UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS BIOLÓGICAS
PROGRAMA DE PÓS GRADUAÇÃO EM BIOINFORMÁTICA
DISSERTAÇÃO DE MESTRADO

IGOR DE BARROS RIGUEIRA FERNANDES

AS ENZIMAS NEGLIGENCIADAS ASSOCIADAS À DEGRADAÇÃO DA LIGNINA EM FUNGOS

Belo Horizonte
2020

Igor de Barros Rigueira Fernandes

AS ENZIMAS NEGLIGENCIADAS ASSOCIADAS A DEGRADAÇÃO DA LIGNINA EM FUNGOS

Dissertação apresentada ao programa de interunidades de Pós-Graduação em Bioinformática da Universidade Federal de Minas Gerais como requisito parcial para obtenção do título de Mestre em Bioinformática.

Orientador: Prof. Dr. Aristóteles Góes-Neto.

Coorientador: Dr. Rodrigo Bentes Kato.

Belo Horizonte
2020

043 Fernandes, Igor de Barros Rigueira.
As enzimas negligenciadas associadas a degradação da lignina em fungos [manuscrito] / Igor de Barros Rigueira Fernandes. – 2020.

277 f. : il. ; 29,5 cm.

Orientador: Prof. Dr. Aristóteles Góes-Neto. Coorientador: Dr. Rodrigo Bentes Kato.
Dissertação (mestrado) – Universidade Federal de Minas Gerais, Instituto de Ciências Biológicas. Programa Interunidades de Pós-Graduação em Bioinformática.

1. Biologia Computacional. 2. Lignina. 3. Enzimas. 4. Fungos. 5. Domínios Proteicos. 6. Modelos Moleculares. I. Góes-Neto, Aristóteles. II. Kato, Rodrigo Bentes. III. Universidade Federal de Minas Gerais. Instituto de Ciências Biológicas. IV. Título.

CDU: 573:004

Ficha catalográfica elaborada pela bibliotecária Rosilene Moreira Coelho de Sá – CRB 6 - 2726

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto De Ciências Biológicas
Programa Interunidades de Pós-Graduação em Bioinformática da UFMG

ATA DE DEFESA DE DISSERTAÇÃO

IGOR DE BARROS RIGUEIRA FERNANDES

Às quatorze horas do dia 10 de março de 2020, reuniu-se, no Instituto de Ciências Biológicas da UFMG, a Comissão Examinadora de Dissertação, indicada pelo Colegiado do Programa, para julgar, em exame final, o trabalho do discente Igor de Barros Rigueira Fernandes, intitulado: **"AS ENZIMAS NEGLIGENCIADAS ASSOCIADAS A DEGRADAÇÃO DA LIGNINA EM FUNGOS"**, requisito para obtenção do grau de Mestre em **Bioinformática.** Abrindo a sessão, o Presidente da Comissão, **Dr. Aristóteles Góes Neto**, após dar a conhecer aos presentes o teor das Normas Regulamentares do Trabalho Final, passou a palavra ao candidato, para apresentação de seu trabalho. Seguiu-se a arguição pelos Examinadores, com a respectiva defesa do candidato. Logo após, a Comissão se reuniu, sem a presença do candidato e do público, para julgamento e expedição de resultado final. Foram atribuídas as seguintes indicações:

Prof./Pesq.	**Instituição**	**Indicação**
Dr. Aristóteles Góes Neto	UFMG	Aprovado
Dr. Vasco Ariston de Carvalho Azevedo	UFMG	Aprovado
Dra. Fernanda Badotti	CEFET-MG	Aprovado
Dr. Daniel Santana de Carvalho	UFMG	Aprovado
Dr. Rodrigo Bentes Kato	UFMG	Aprovado

Pelas indicações, o candidato foi considerado: **Aprovado**

O resultado final foi comunicado publicamente ao candidato pelo Presidente da Comissão. Nada mais havendo a tratar, o Presidente encerrou a reunião e lavrou a presente ATA, que será assinada por todos os membros participantes da Comissão Examinadora.

Belo Horizonte, 10 de março de 2020.

Documento assinado eletronicamente por **Aristoteles Goes Neto, Professor do Magistério**

Superior, em 07/01/2021, às 18:10, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Rodrigo Bentes Kato, Usuário Externo**, em 11/01/2021, às 09:25, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Daniel Santana de Carvalho, Usuário Externo**, em 12/01/2021, às 14:43, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Fernanda Badotti, Usuário Externo**, em 14/01/2021, às 18:10, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Vasco Ariston de Carvalho Azevedo, Professor do Magistério Superior**, em 26/01/2021, às 11:01, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0501589** e o código CRC **444E7A62**.

Referência: Processo nº 23072.200211/2021-35 **SEI nº 0501589**

AGRADECIMENTOS

Gostaria de agradecer a Deus que guia meus caminhos, diante de tantas coincidências me proporcionando os recursos necessários para realizar esse trabalho e quando diante do cansaço me fez forte para sempre continuar em frente.

Agradeço ao meu orientador Professor Aristóteles Góes-Neto, que me aceitou no grupo de pesquisa e coorientador Rodrigo Kato, que me indicou o grupo de pesquisa, ambos acreditaram no meu potencial, compartilharam seu tempo e conhecimento, sempre entendendo minhas dificuldades pessoais e fizeram de tudo para me ajudar o máximo possível.

A professora Glória Franco que além de ser uma excelente professora me indicou o mestrado na UFMG e me auxiliou nos momentos mais simples aos mais complicados que passei.

A professora Mariana Magalhães que foi minha professora em muitas disciplinas, além de me ouvir e auxiliar em situações que não tinha a quem recorrer.

Ao professor Francisco Lobo por ter sido um grande professor e por me disponibilizar seu programa, além de ter tido a paciência em me auxiliar em dúvidas e dificuldades de sua execução.

A todos do lbmcf (Laboratório de Biologia Molecular e Computacional de Fungos) formado por alunos fantásticos e cooperativos, que sempre estão aberto ao diálogo e no compartilhamento de informação.

A Sheila Santana, Tiago Silva e todos os membros da Secretaria do Programa de Pós-Graduação em Bioinformática por toda dedicação, simpatia e atenção, sempre me auxiliando nos problemas e burocracias de forma ágil e eficiência inigualável.

A todos os amigos que tive o prazer de conhecer durante o curso e que me apoiaram com palavras e companheirismos me ajudando a enfrentar os piores desafios.

A todos maravilhosos professores que conheci durante meu mestrado, que dedicaram seu tempo a me tornar melhor e capaz, pois sabem que a educação é a maior ferramenta para mudar a realidade de alguém.

Ao meu pai Alexandre Fernandes, que sempre esteve me aconselhando nas dificuldades e me apoiando financeiramente e à minha mãe Edineia Rigueira que sempre me escutou e me acudiu nos momentos difíceis. Ambos me deram amor incondicional, se dedicaram a me tornar melhor e fizeram de tudo para chegar onde cheguei.

À minha namorada Dandara Garlet, que sempre me dedicou carinho e atenção, me apoiando e compreendendo todas minhas dificuldades, me auxiliando sempre que podia, sendo além de minha maravilhosa companheira uma inestimável amiga.

À toda Universidade Federal de Minas Gerais que é composta pelos melhores profissionais e mais dedicados pesquisadores e professores que conheço.

Muito obrigado a todos.

RESUMO

Uma alternativa como fonte de recursos renováveis para a produção de bioenergia é a utilização de material lignocelulósico, derivado de resíduos agroindustriais, porém esses processos ainda apresentam dificuldades de implementação devido ao alto custo, sendo que os métodos enzimáticos apresentam grande potencial para contornar essas dificuldade, mas ainda faltam informações para o desenvolvimento de processos enzimáticos eficientes. Para isso, é muito importante investigar em detalhes, além das principais enzimas envolvidas na sua decomposição (lignina peroxidases, manganês peroxidases, versátil peroxidases e lacases), as enzimas que contribuem indiretamente, mas são também muito importantes, para o processo como um todo. Essa dissertação abordou justamente esse tema: as enzimas negligenciadas do metabolismo de degradação de lignina em fungos. O banco de dados CAZy é uma referência curadas de enzimas envolvidas na degradação de carboidratos de uma maneira geral. Como a lignina está fortemente associada aos carboidratos das paredes celulares de vegetais, há uma categoria nesse banco denominada de AuxiliaryActivities (AA), que são enzimas que atuam direta ou indiretamente na degradação da lignina. Utilizando os dados contidas no CAZy foi construído um banco de sequências e taxonomia de cada sequência. Com este banco de dados customizado de sequências e taxonomia de AA de fungos, fez-se uma mineração dos dados. Fez-se também a análise dos domínios pelo Pfam e correlacionou-se com o alinhamento das sequências de cada família a fim de associar esses dados, obtendo-se informações diversas, como a distribuição taxonômica dos organismos que continham essas sequências, a identificação dos domínios proteicos conservados e a obtenção de padrões estruturais, além da identificação inesperada de mecanismos prováveis até então não descritos na literatura. Além disso, buscando evidências da importância dessas enzimas para os organismos que as possuem, realizou-se uma análise de seleção positiva com a identificação de possíveis grupos ou sítios onde se têm evidência. A construção de modelos moleculares tridimensionais, permitiu ainda associar dados estruturais à literatura, e esses modelos 3D mostraram que as regiões selecionadas tem grande potencial de aumentar a estabilidade estrutural, evitando a desnaturação da enzima.

Palavras-chave: Lignina, seleção positiva, enzimas ligninolíticas negligenciadas, Fungi, domínios proteicos.

ABSTRACT

An alternative as a source of renewable resources for the production of bioenergy is the use of lignocellulosic material, derived from agro-industrial residues, however these processes still present implementation difficulties due to the high cost, and enzymatic methods have great potential to overcome these difficulties, but information is still lacking for the development of efficient enzymatic processes. For this, it is very important to investigate in detail, in addition to the main enzymes involved in its decomposition (lignin peroxidases, manganese peroxidases, versatile peroxidases and laccases), the enzymes that contribute indirectly, but are also very important, to the process as a whole. This dissertation addressed precisely this theme: the neglected enzymes of the metabolism of lignin degradation in fungi. The CAZy database is a cured reference for enzymes involved in the breakdown of carbohydrates in general. As lignin is strongly associated with carbohydrates in the cell walls of vegetables, there is a category in this bank called Auxiliary Activities (AA), which are enzymes that act directly or indirectly on the degradation of lignin. Using the data contained in CAZy a bank of sequences and taxonomy of each sequence was built. With this customized database of fungi AA sequences and taxonomy, data mining was carried out. Domains were also analyzed by Pfam and correlated with the alignment of the sequences of each family in order to associate these data, obtaining diverse information, such as the taxonomic distribution of the organisms that contained these sequences, the identification of the domains conserved proteins and the achievement of structural patterns, in addition to the unexpected identification of probable mechanisms hitherto not described in the literature. In addition, looking for evidence of the importance of these enzymes for the organisms that have them, a positive selection analysis was carried out with the identification of possible groups or sites where there is evidence. The construction of three-dimensional molecular models, also allowed to associate structural data to the literature, and these 3D models showed that the selected regions have great potential to increase structural stability, avoiding denaturation of the enzyme.

Keywords: Lignin, positive selection, neglected ligninolytic enzymes, Fungi, protein domains.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIAÇÕES

CAZy	*Carbohydrate Active EnzymeDatabase*
AA	*AuxiliaryActivities*
NGS	Sequenciamento de próxima geração
LiP	Peroxidases de lignina
MnP	Peroxidases de manganês
VP	Peroxidases versáteis
VA	Álcool veratrílico
V	Volts
Sub	Subfamília
CDH	Celobiose desidrogenases
AAO	Aril-álcool oxidase
GOX	Glucose 1-oxidase
AOX	Álcool oxidase
POX	Piranose oxidase
CBM	Domínio de ligação à carboidratos
CBD	Domínio de ligação à celulose
VAO	Vanilil-álcool oxidases
GLOX	Glioxal oxidases
GAO	Galactose oxidases
AO	Álcool oxidase
BQR	1,4-Benzoquinone redutase
FMN	Mononucleotídeo de flavina
QR	Quinonas redutases
FAD	Dinucleótido de flavina e adenina
NADH	Dinucleótido de nicotinamida e adenina
NADPH	Fosfato de dinucleótido de nicotinamida e adenina
GOOX	Glucooligossacarídeo oxidase
CHITO	Quitooligossacarídeo oxidase
DRF	Domínio de redutase de ferro
LPMO	Polissacarídeo lítico monooxigenases
PDH	Piranose desidrogenase

PQQ	PirroloquinolinaQuinona
EC	*EnzymeCommissionNumbers*
ID	Identificação
PDB	*Protein Data Bank*
NCBI	*National Center for BiotechnologyInformation*
enzBBE	*Berberine Bridge Enzyme*
CBQ	Celobiosequinonaoxidoredutase
CBO	Celobiose oxidase
DNA	*DeoxyribonucleicAcid*
WGS	*Wholegenomesequencing*

SUMÁRIO

1. INTRODUÇÃO

1.1. Importância e desafios do processamento da biomassa

As paredes das plantas apresentam lignina, estando ligada com a celulose e hemicelulose por meio de ligações covalentes e de hidrogênio, tornando a estrutura resistente contra a degradação (DE GONZALO *et al.*, 2016). A madeira e resíduos agrícolas são as matérias primas mais desejadas como fonte de recurso natural para produção de combustíveis. A biomassa é um recurso largamente encontrado em muitas áreas do mundo, sendo possível a exploração de forma sustentável (GHOREISHI *et al.*, 2019). Além disso a silvicultura pode aumentar a disponibilidade sem competir com a produção de alimentos, contrastando fortemente com o aumento de uso de óleos vegetais e açúcar, uma vez que o uso da madeira como recurso renovável evita os efeitos colaterais das culturas agrícolas intensivas como soja, cana-de-açúcar e canola (STEVENS *et al.*, 2019). A lignina também é a única matéria prima renovável de grande volume que apresenta monômeros aromáticos, servindo como fonte desse tipo de material, podendo ser um substituto alternativo de compostos aromáticos de recursos fósseis (ISIKGOR *et al.*, 2015). A importância da utilização da biomassa como fonte de recursos renováveis pode ser vista na Lei de Segurança e Independência Energética dos EUA de 2007 que pretende até 2022 estimular o desenvolvimento da capacidade de produção de 79 bilhões de litros de biocombustíveis de segunda geração anualmente (RAGAUSKAS *et al.*, 2014).

Para uma eficiente conversão da celulose e hemicelulose em açúcares fermentados é necessário realizar o pré-tratamento da biomassa de lignocelulose. Esse pré-tratamento busca reduzir o tamanho das partículas de biomassa, modificar sua estrutura ou até sua composição química a fim de tornar a celulose mais acessível a etapa de hidrólise (LI *et al.*., 2013). Dessa forma o pré-tratamento permite a conversão dos compostos sacarídeos de forma mais eficiente e rápida, sendo o processo de pré-tratamento a etapa que mais exige energia durante a conversão de biomassa em biocombustíveis (DUQUE *et al.*, 2016). Desse modo, a escolha da técnica utilizada como pré-tratamento leva em consideração a taxa de hidrólise buscando aumentar a eficiência, não gerar inibidores da fermentação, ter baixo custo, preservar os carboidratos, permitir a recuperação da lignina e baixo consumo de energia (DA SILVA *et al.*, 2013).

Os processos de pré-tratamento podem ser físicos, como lascagem, moagem, microondas, ultrassom e explosão de vapor. Esses processos físicos podem ser considerados caros energeticamente e exigem maquinário complexo e monitoramento rigoroso, além disso

técnicas, como explosão a vapor, podem gerar inibidores enzimáticos e fermentativos que incluem furfural e hidroximetilfurfural, além de ácidos fracos como ácidos acético, fórmico e levulínico (DA SILVA *et al.*, 2013). Também existem processos químicos, por exemplo, processos que utilizam ácidos e bases fortes, porém estes apresentam desvantagens como no caso dos ácidos que podem corroer as estruturas, necessitando de equipamentos resistentes e mais caros, além da poderem produzir inibidores e apresentar toxicidade. Já os processos utilizando bases são baratos e eficientes, porém necessitam da neutralização da base usada, produzindo sais que podem permanecer na biomassa, além de ser um processo mais lento que demanda instalações maiores (SINGH *et al.*, 2015).

Existe os pré-tratamento biológico que, diferentemente das demais, não produz resíduos químicos, sendo um tratamento ecológico e amigável para o meio ambiente, além de não gerar inibidores de fermentação e permitir uma elevada preservação das cadeias polissacarídicas e consumirem uma quantidade reduzida de energia durante o processo (MORENO *et al.*, 2015). Umas das formas para realizar esse tratamento utiliza o crescimento microbiológico, principalmente de fungos, que tem como função liberar enzimas para degradar a lignina. O uso de microorganismos ainda enfrentam dificuldades como o longo tempo de processamento, necessidade de grandes áreas, adições de nutrientes como nitrogênio, Mn^{+2} e Cu^{+2}, além do monitoramento constante do crescimento do microrganismo e das condições de crescimento, como pH e temperatura. Esses problemas inviabilizam o uso do processamento biológico (HAGHIGHI MOOD *et al.*, 2013; MORENO *et al.*, 2015).

Como um pré-tratamento biológico, também existe a possibilidade do uso de enzimas ligninolíticas purificadas ao invés do uso dos microorganismos. Essa estratégia permite aumentar a eficiência da despolimerização, reduzindo o tempo de reação para algumas horas ao invés de semanas como é no caso do uso dos microorganismos. O uso de enzimas também diminui a necessidade de suplementação de nutrientes e permite que se possa trabalhar em uma faixa aumentada de pH e temperatura (MORENO *et al.*, 2015; POLIZELI *et al.*, 2013). Os principais grupos de enzimas utilizadas na despolimerização da lignina são as peroxidases e as laccases. A utilização de enzimas pode ser inviabilizada pelo preço da produção das enzimas e fatores que dificultam a utilização. Como exemplo, as laccases que apresentam atividade na porção fenólica da lignina e necessitam de mediadores para conseguir degradar a porção não-fenólica. Além disso no caso das peroxidases, necessitam da presença de peróxido de hidrogênio para realizar a degradação, também podendo apresentar necessidade de alguns mediadores. Esses são alguns dos fatores que inviabilizam o atual uso de enzimas lignolíticas,

necessitando o desenvolvimento de novas abordagens tecnológicas para tornar os processos enzimáticos viáveis (CHEN *et al.*, 2012; MORENO *et al.*, 2015; POLIZELI *et al.*, 2013).

1.2. Lignina

A lignina é um polímero essencial para a vida das plantas vasculares, permitindo maior rigidez da parede celular e resistência a infecção por patógenos (BHUIYAN *et al.*, 2009; PEREIRA *et al.*, 2018). A lignina protege as cadeias de celulose, dificultando a sua degradação. A biossíntese ocorre a partir da polimerização principalmente de três unidades básicas que são os álcoois coniferil, sinapil e p-coumaril (POLLEGIONI *et al.*, 2015). A biossíntese desses monolignóis é feita a partir do aminoácido fenilalanina e a polimerização da estrutura da lignina ocorre pela oxidação dos grupos hidroxilas desses monolignóis. A desidrogenação enzimática dos monolignóis produz espécies reativas que acabam se acoplando entre si. Diversas reações são possíveis, fazendo com que se forme uma estrutura amorfa (BHUIYAN *et al.*, 2009; HIGUCHI, 1990).A estrutura da lignina é praticamente não-fenólica, apesar dos seus componentes primários serem fenóis. Isto ocorre porque durante a formação da lignina há um predomínio da formação de éteres, fazendo com que a estrutura seja predominantemente não-fenólica, com poucos componentes fenóis (RUIZ-DUEÑAS *et al.*, 2009).

A lignina é a barreira de defesa contra a degradação da celulose, impedindo que enzimas consigam ter contato com a cadeia polissacarídica e a degrademem monômeros disponíveis para consumo energético pelas células. Ela também se associa com a hemicelulose, próximo à estrutura da celulose (GRABBER, 2005). Isso é um problema, por exemplo, na obtenção de etanol de segunda geração, uma vez que impede que os açúcares fiquem disponíveis para serem metabolizados pelas leveduras (PONNUSAMY *et al.*, 2019).

1.3. Dados biológicos e desafios relacionados à análise

O surgimento da tecnologia de sequenciamento massivamente paralelo ou sequenciamento de próxima geração (NGS) criou uma revolução nas análises genômicas, fazendo com que grandes quantidades de dados começassem a ser produzido. Porém, independente da quantidade de dados gerados, esses precisam ser processados para que possam produzir conhecimento e descobertas (MUIR *et al.*, 2016). Devido a necessidade de processamento dos dados surgiu um novo desafio relacionado à capacidade de análise, visto que o crescimento dessa capacidade de análise é menor em comparação a quantidade de dados gerados anualmente (SBONER *et al.*, 2011).

Uma forma de facilitar as análises é a organização dos dados em bancos de dados, permitindo um acesso rápido aos cientistas as informações de interesse, diminuindo a necessidade de reavaliar grandes massas de dados a cada novo projeto (MUIR *et al.*, 2016). Um problema decorrente aos bancos de dados é a garantia de precisão das sequências dentro dos bancos. Antes do aumento do número de sequenciamentos, os dados acrescentados nos bancos passavam por análises completas como as de bancada, que permitiam maior curadoria evitando os falsos positivos (BENGTSSON-PALME *et al.*, 2016). Atualmente devido à grande quantidade de novos sequenciamentos muitas das informações geradas são obtidas por análises computacionais automatizadas, que podem gerar grandes quantidades de erro de anotação. Esses erros podem acarretar em mais erros futuros, caso sequências identificadas erroneamente forem utilizadas para identificar novas sequências. Dessa forma diversas estratégias são desenvolvidas buscando reavaliar os dados curados automaticamente buscando uma melhora contínua (SCHNOES *et al.*, 2009).

Outra forma de contornar os desafios computacionais é a criação e utilização de softwares mais rápidos e eficientes. Muito esforço é dedicado na construção de programas, utilizando estratégias da ciência da computação voltada para a construção de ferramentas de análise de dados biológicos (MUIR *et al.*, 2016). Uma das principais técnicas utilizadas e que vem evoluindo rapidamente são as relacionadas ao alinhamento de sequências.

Diversas ferramentas e tipos de análise dependem de técnicas de alinhamento, indo desde a comparação de sequências curtas contra um banco de dados como ocorre pela ferramenta BLAST (ALTSCHUL *et al.*, 1990) ou o Diamond (BUCHFINK *et al.*, 2015), como alinhadores como o MAFFT (ROZEWICKI *et al.*, 2019) e diversos montadores de genomas, que alinham os fragmentos obtidos do sequenciamento em busca de decifrar a ordem correta como se fosse um quebra-cabeça (MUIR *et al.*, 2016).

Existe também programas como o HMMer (HMMER, 2019) que utiliza modelos probabilísticas de cadeias ocultas de Markov para analisar sequências utilizando modelos previamente estabelecidos característicos de uma sequência ou região de interesse. Dessa forma é possível a utilização de modelos característicos para identificação de regiões com grau de conservação, permitindo a identificação eficiente de domínios conservados (EDDY, 2019). O Pfam é um programa que utiliza o HMMer como uma das etapas de identificação de domínios, comparando as sequências a serem analisadas com um banco de dados próprio de regiões conservadas e curadas pelos desenvolvedores (FINN *et al.*, 2014). Essas

características de domínios podem ser utilizadas para caracterizar sequências de um banco de dados, evitando falso positivos (LOMBARD *et al.*, 2014).

1.4. Carbohydrate Active EnzymeDatabase (CAZy)

O banco de dados *Carbohydrate Active EnzymeDatabase* (CAZy), consiste em uma base de classificação de sequências proteicas ativas em carboidratos que existe desde 1998. O CAZy foi construído devido à complexidade de enzimas envolvidas em reações que utilizam carboidratos. A quantidade de conformações envolvendo carboidratos é grande, devido à complexidade estereoquímica e diferentes formas de combinações dos seus grupos hidroxila (LOMBARD *et al.*, 2014). Dessa forma os desenvolvedores do banco de dados buscaram uma forma eficiente de classificar as enzimas envolvidas na modificação, formação e quebra dos componentes dos carboidratos (LEVASSEUR *et al.*, 2013; LOMBARD *et al.*, 2014). Para isso os desenvolvedores do banco de dados realizaram comparação de sequências de aminoácidos, conformação da estrutura proteica e mecanismo de reação para criar classificações eficientes. Os dados ficam disponíveis em forma de tabelas contendo alguns metadados, entre eles, o código de depósito e o banco em que está depositado a sequência proteica.

Atualmente o CAZy é composto por cinco classes que abrigam mais de trezentas famílias enzimáticas. As classes são *glycosidehydrolases*(GH), *glycosyltransferases* (GT), *polysaccharidelyases* (PL), *carbohydrateesterases* (CE), e *AuxiliaryActivities* (AA). O banco de dados CAZy é atualizado constantemente com sequências curadas e metadados dessas sequências, além de conter informações organizadas de estruturas tridimensionais do *Protein Data Bank* (PDB) (BERMAN, 2000; LEVASSEUR *et al.*, 2013; LOMBARD *et al.*, 2014). A importância da lignina na estruturação da lignocelulose fez com que o banco CAZy criasse uma categoria com o nome AA, a qual abrange enzimas envolvidas direta ou indiretamente na degradação da lignina. Atualmente, a categoria AA é dividida em 16 subcategorias chamadas de famílias e agrupa enzimas envolvidas desde a ação direta da despolimerização da lignina, até enzimas acessórias que produzem peróxido de hidrogênio auxiliando a função das enzimas peroxidases (LEVASSEUR *et al.*, 2013).

1.5. Fungos degradadores de biomassa

Os microorganismos são grandes degradadores de material lignocelulótico, porém os fungos tendem a ser mais eficientes nessa tarefa se comparado com as bactérias. Dessa forma a degradação da biomassa por bactérias é mais lenta, além da lignina não ser atacada pela

maioria dessas bactérias decompositoras de material lignocelulósico (DANIEL, 2003; ERIKSSON *et al.*, 1990; TAHA *et al.*, 2015). Os Basidiomycota e Ascomycota são os principais filos de fungos degradadores de biomassa, sendo geralmente classificados entre fungos de podridão macia, podridão marrom e podridão branca (MADADI; ABBAS, 2017; MARTÍNEZ *et al.*, 2005).

Os fungos de podridão macia são formados principalmente por Ascomycota e conseguem atacar eficientemente o material polissacarídeo da biomassa, sendo pouco eficiente na degradação da lignina. O ataque limitado pelos fungos da podridão mole geram um aspecto mole para a madeira quando colocada em ambiente úmido ou quebradiço quando em ambiente seco. A atividade dos fungos de podridão macia na decomposição da biomassa é lenta comparada às podridões branca e marrom, resultante da baixa atividade de despolimerização da lignina (ERIKSSON *et al.*, 1990; SIGOILLOT *et al.*, 2012).

Os fungos de podridão marrom crescem principalmente em madeiras macias, podendo degradar os polímeros polissacarídeos, após uma modificação parcial da lignina (BUGG *et al.*, 2011). Os fungos de podridão marrons têm uma capacidade reduzida de degradar lignina em comparação aos fungos de podridão branca, apresentando mecanismo de despolimerização da lignina por reações de Fenton, realizando uma leve oxidação da lignina e resultando em uma fonte potencial de compostos aromáticos para a fração estável de matéria orgânica em solos das florestas. O composto incompleto produzido no final da degradação apresenta cor amarronzada que originou o nome da classificação que engloba esses fungos (SIGOILLOT *et al.*, 2012).

Os fungos de podridão branca têm capacidade de degradar eficientemente todos os componentes da madeira como celulose, hemicelulose e lignina. A extensa degradação da biomassa faz com que a região degradada apresente uma cor esbranquiçada (SIGOILLOT *et al.*, 2012). Os Basidiomycetos são os principais degradadores de lignina, sendo os Basidiomycetos classificados como os principais fungos de podridão branca e considerados os mais eficientes degradadores de lignina (SÁNCHEZ, 2009; SIGOILLOT *et al.*, 2012). Os fungos de podridão branca apresentam uma gama de enzimas que atacam diretamente a lignina como lacases e peroxidases ligninolíticas, sendo essas enzimas responsáveis pela alta eficiência da despolimerização da lignina (BUGG *et al.*, 2011).

1.6. Enzimas degradadoras de lignina e sua importância

Existem quatro principais enzimas que realizam a degradação da lignina: peroxidases de lignina (LiP) (EC 1.11.1.14), peroxidases de manganês (MnP) (EC 1.11.1.13), peroxidases

versáteis (VP) (EC 1.11.1.16) e lacases (EC 1.10.3.-). A LiP é uma heme proteína, peroxidase de classe II que é secretada durante o processo de degradação fúngica da madeira (DOYLE *et al.*, 1998). A LiP apresenta um elevado potencial redox de cerca de 1.4 V, o qual garante à LIP a capacidade de oxidar compostos fenólicos e não-fenólicos sem a necessidade de mediador. A reação enzimática ocorre pela oxidação do Fe (III) da heme pelo peróxido de hidrogênio, transferindo dois elétrons e fazendo com que o Fe se transforme no composto I [Fe(IV) = O^{+}] com a redução de peróxido de hidrogênio. Depois o composto I oxida um substrato doador, se transformando inicialmente no composto II [Fe (IV) = O] e gerando o substrato reduzido. Por fim, uma nova oxidação do substrato doador faz com que a LiP retorne para seu estado de repouso (BUGG *et al.*, 2011). Quando em excesso de peróxido de hidrogênio, o composto II é oxidado novamente gerando o composto III [Fe(IV) = O_2^{-}], podendo ser recuperado pela transferência de elétrons para um substrato doador. Como aceptor de elétrons, a LiP utiliza-sedoálcool veratrílico (VA), uma molécula que pode ser sintetizada pelos fungos a partir da fenilalanina (JENSEN *et al.*, 1994; SHIMADA *et al.*, 1981).

A MnP é uma proteína heme glicosilada, sendo a mais comum peroxidase modificadora de lignina produzida, contida em quase todos os fungos basidiomicetos (CHEN *et al.*, 2012). A MnP utiliza de Mn para oxidar a lignina. A reação começa com o peróxido de hidrogênio interagindo com o Fe da enzima e gerando o composto I [Fe (IV) = O^{+}]. Depois de oxidado a enzima interage com o Mn^{2+} transformando-o em Mn^{3+}. O Mn^{3+} é então quelado por ácidos orgânicos e pode ser difundido pelo meio até a lignina. Duas etapas de redução são necessárias para o Fe retornar ao seu estado nativo. Na primeira etapa de redução o Fe forma o composto II [Fe (IV) = O] e na segunda etapa retorna ao estado nativo. O excesso de peróxido de hidrogênio pode fazer com que o Fe seja oxidado de forma reversível para o composto III [Fe (IV) = O_2^{-}], podendo ser reduzido através da oxidação de um átomo de Mn^{2+}. O complexo (Mn^{3+} com o quelante) oxida compostos fenólicos por retirada de hidrogênios, gerando radicais fenoxil (HOFRICHTER, 2002).

A VP é uma enzima que apresenta arquitetura híbrida, apresentando diferentes sítios de oxidação que interagem com o cofator heme. Ela tem o sítio de interação da LiP, podendo degradar a lignina não-fenólica, utilizando a oxidação do VA para aumentar sua capacidade redox. A VP também tem uma estrutura semelhante ao sítio de interação da MnP, podendo oxidar Mn^{2+} gerando reagentes difusíveis. O mecanismo de reação da VP é semelhante aos mecanismos de reação da LiP e da MnP (PÉREZ-BOADA *et al.*, 2005; SRIDHAR, 2016).

As lacases são enzimas fenol oxidase que contém 4 íons de cobre na sua estrutura. As lacases estão amplamente distribuídas entre os organismos vivos, presente em plantas, insetos, bactérias e fungos. A lacase tem a capacidade de oxidar uma ampla diversidade de substâncias e para isso utiliza-se da clivagem da molécula de oxigênio molecular, gerando água como subproduto (GIARDINA *et al.*, 2010). As lacases tem capacidade de oxidar os compostos fenólicos da lignina gerando radicais fenoxil. Foi observado que elas conseguem degradar lignina não-fenólica na presença de mediadores como ABTS ou ácido 3-hidroxiantranílico (CHEN *et al.*, 2012).

As enzimas LiP, MnP e VP necessitam de peróxido de hidrogênio para conseguirem degradar a lignina. Para a ação dessas peroxidases, enzimas acessórias permitem a geração constante de peróxido de hidrogênio (DASHTBAN *et al.*, 2010).Como, por exemplo, a enzima aril álcool oxidase que, durante a redução de álcoois para aldeído, produz peróxido de hidrogênio, e a enzima glioxal oxidase que cataliza a oxidação de aldeídos para ácidos carboxílicos e produz peróxido de hidrogênio (DASHTBAN *et al.*, 2010). Além disso as enzimas acessórias realizam outras atividades que auxiliam as enzimas lignolíticas como novamente a enzima glioxal oxidase, que além de produzir peróxido de hidrogênio, também consegue produzir ácidos orgânicos que auxiliam como quelante do Mn^{3+} (URZÚA *et al.*, 1998) e também enzimas como a 1,4-benzoquinone reductases capazes de catalisar a redução dos compostos fenólicos gerados durante a degradação da lignina impedindo que voltem a se repolimerizar e estão envolvidas na melhoria do metabolismo de moléculas oriundas da degradação da lignina pelos fungos (MORENO *et al.*, 2015; MORI *et al.*, 2016). Mesmo depois de descrito a importância das enzimas acessórias sabemos que elas não recebem a mesma atenção quanto o que é dada para as enzimas lignolíticas da classe das lacases e das peroxidases. Uma prova disso é a inexistência de estruturas para a glioxal oxidase até o período que esse trabalho está sendo escrito, ou mesmo o número de resultados muito inferiores de trabalho encontrados em comparação as enzimas lignolíticas. Dessa forma traremos na próxima seção uma breve revisão das enzimas acessórias contidas nas famílias AA3 até a AA16, a fim de descrever a importância e a relação dessas enzimas com a degradação da lignina. Essas enzimas acessórias estão contidas no banco de dados CAZy. Utilizamos o CAZy como referência de sequências por se tratar de um banco curado.

1.7. Enzimas negligenciadas associadas a degradação da lignina: famílias AA3 até AA16

1.7.1. Família AA3

Essas enzimas pertencem a família da glicose-metanol-colina, sendo flavoproteínas, apresentando o sítio de ligação de flavina-adenina. Essas enzimas têm como característica, produção de peróxido de hidrogênio como um dos subprodutos das suas reações. Essa família está dividida em quatro subfamílias (sub), e são constituídas por AA3sub1 celobiose desidrogenases (CDH), AA3sub2 aril-álcool oxidase (AAO) e glucose 1-oxidase (GOX), AA3sub3 álcool oxidase (AOX) e AA3sub4 piranose oxidase POX (LEVASSEUR *et al.*, 2013).

1.7.1.1. Celobiose desidrogenase (EC 1.1.99.18)

A celobiose desidrogenase (CDH) é uma enzima extracelular envolvida no metabolismo de carboidratos. É uma flavocitocromo enzima e consegue metabolizar carboidratos como celobiose e manobiose para lactonas, transferindo dois elétrons para uma quinona, radical fenoxil ou um oxigênio molecular. Celobiose e manobiose são carboidratos produzidos durante a degradação da biomassa vegetal por fungos (HENRIKSSON *et al.*, 2000; PERTILE *et al.*, 2019). Como subproduto da reação a CDH pode gerar peróxido de hidrogênio que auxiliaria diversas enzimas peroxidases vistas anteriormente. Ela também está envolvida na reação de redução do Fe III para Fe II, que pode ocorrer no lugar de geração do peróxido de hidrogênio (Figura 1) (HENRIKSSON *et al.*, 2000). A CDH é composta por múltiplos domínios, sendo o domínio citocromo do tipo b ligado a um grande domínio da flavo desidrogenase. Um domínio de ligação à celulose (CBD) também pode estar presente na estrutura da enzima, ou mesmo uma sequência rica em aminoácidos aromáticos sendo responsáveis pela interação da CDH na celulose (HALLBERG *et al.*, 2000; HENRIKSSON*et al.*, 2000).

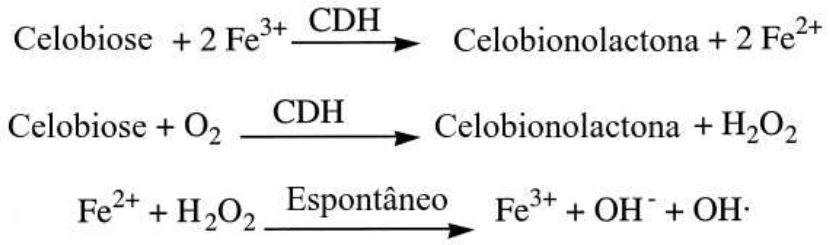

Figura 1 - Equação de reação da enzima CDH e a reação de Fenton. Fonte: Adaptada de (HENRIKSSON et al., 2000).A figura acima mostra os reagentes e os produtos gerados durante a reação enzimática com a CDH. Ela também mostra a reação de fenton que recupera o estado de oxidação do Fe III que permite uma concentração de Fe III para o funcionamento da primeira etapa da figura.

1.7.1.2. Aril-álcool oxidase (EC 1.1.3.7)

A enzima aril-álcool oxidase (AAO) é uma flavoproteína presente em muitos fungos, principalmente basidiomicetos. Ela está sendo estudada devido a sua relação com a degradação de lignina por fungos, sendo capaz de oxidar álcoois e gerar como um dos seus subprodutos peróxido de hidrogênio (Figura 2). A AAO tem uma ampla especificidade de substratos, podendo atuar principalmente nos fenóis gerados pela degradação da lignina (HERNÁNDEZ-ORTEGA *et al.*, 2012). A enzima AAO é extracelular e no seu mecanismo é utilizado oxigênio molecular e fenóis provenientes da degradação da biomassa. Um dos melhores substratos é o álcool 4-metoxibenzóico, reforçando a ideia do envolvimento da AAO na degradação da lignina visto que o álcool 4-metoxibenzóico é um metabólico muito encontrado durante a degradação da lignina. O peróxido de hidrogênio gerado serve como substrato para a ação das peroxidases fúngicas, fazendo com que a AAO seja uma enzima muito importante envolvida na degradação da lignina (GUPTA, 2016).

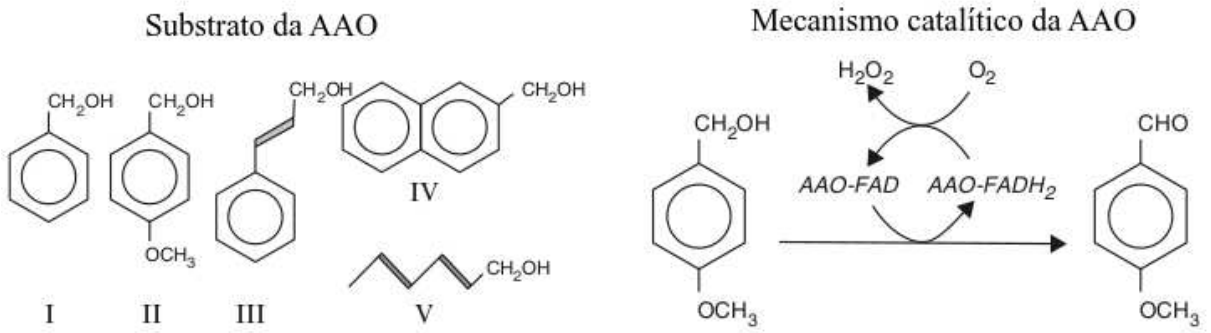

Figura 2 - Rota de ação da enzima AAO, sendo I) álcool benzílico; II) p-anisil álcool; III) álcool cinamílico; IV) 2-naftalenometanol; e V) 2,4-hexadien-1-ol. Também mostra a rota de ação da enzima para a obtenção de peróxido de hidrogênio. Fonte: Retirada e adaptada de (FANG; SMITH, 2016).

1.7.1.3. Glucose 1-oxidase (EC 1.1.3.4)

Glucose 1-oxidase (GOX) é uma importante enzima produtora de peróxido de hidrogênio encontrada em fungos, como os *Aspergillus* e *Penicillium*, além de basidiomicetos. A GOX é uma flavoproteína capaz de catalisar a oxidação do grupo hidroxila na posição C1 dos açúcares, utilizando, para isso, oxigênio molecular, e gerando como subproduto peróxido de hidrogênio e lactona (LEVASSEUR *et al.*, 2013) (Figura 3). A GOX também é aplicada em sensor de glicose e em preservação de alimentos, retirando o oxigênio molecular presente em alimentos, como suco de frutas. A importância da GOX na degradação da lignina é a produção de peróxido de hidrogênio que auxilia as peroxidases na degradação da lignina (BANKAR *et al.*, 2009).

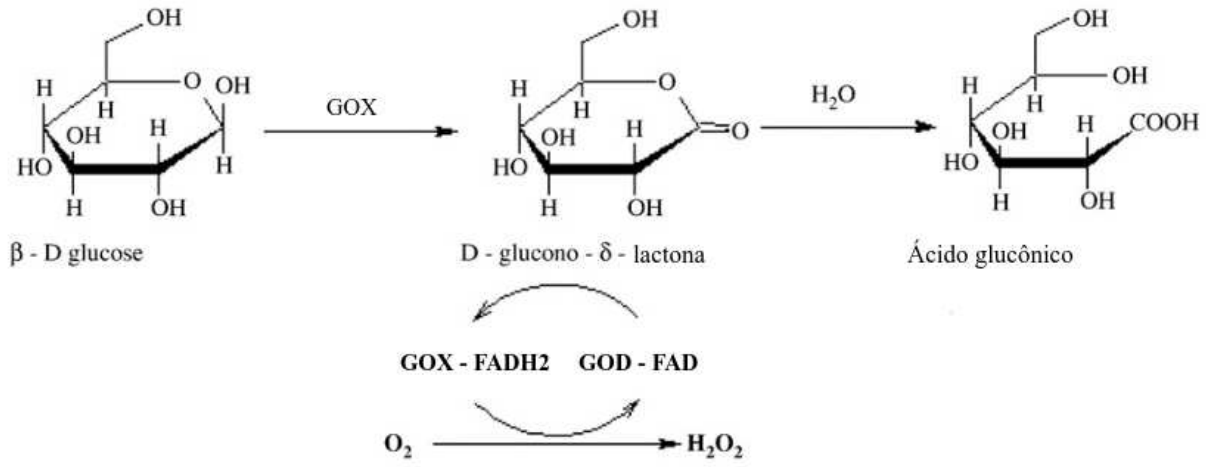

Figura 3 - Rota de ação da enzima GOX. Fonte: Adaptada de (BANKAR et al., 2009).

1.7.1.4. Álcool oxidase (EC 1.1.3.13)

As álcoois oxidases (AOX), são flavoenzimas que oxidam álcoois primários pequenos como metanol e etanol. As estruturas tridimensionais disponíveis nos bancos de dados apresentam o sítio de interação com os substratos reduzidos, comparando com outras álcool oxidases, a região reduzida da enzima pode explicar porque essa enzima reage de forma mais eficiente com álcool com cadeia alifática curta (DANIEL *et al.*, 2007). A enzima utiliza-se de oxigênio molecular presente no meio para oxidar os álcoois, gerando como subprodutos aldeídos e peróxido de hidrogênio (NGUYEN *et al.*, 2018) . A AOX é uma enzima de membranas periplasmáticas e extracelular que consegue produzir peróxido de hidrogênio utilizando provavelmente o metanol proveniente da desmetilação da lignina. Sua ampla presença entre basidiomicetos sustentam a ideia do seu envolvimento na degradação da lignina (KOCH *et al.*, 2016).

1.7.1.5. Piranose oxidase (EC 1.1.3.10)

A piranose oxidase (POX) é uma enzima produtora de peróxido de hidrogênio. A POX é uma flavoenzima extracelular que utiliza oxigênio molecular para catalisar a oxidação do C2 de várias aldopiranoses, preferencialmente glucopiranose (TAN *et al.*, 2014). Como subprodutos a POX gera peróxido de hidrogênio e 2-ceto açúcares correspondentes (Figura 4). A POX é muito estudada devido a sua capacidade de ser utilizadapara produzir açúcares raros, na química fina e em biocélulas de combustíveis (HASSAN *et al.*, 2013). Ela também é amplamente encontrada nos fungos de podridão branca que aponta para sua importância em auxiliar a degradação da lignina pela produção de peróxido de hidrogênio (BANNWARTH *et al.*, 2004).

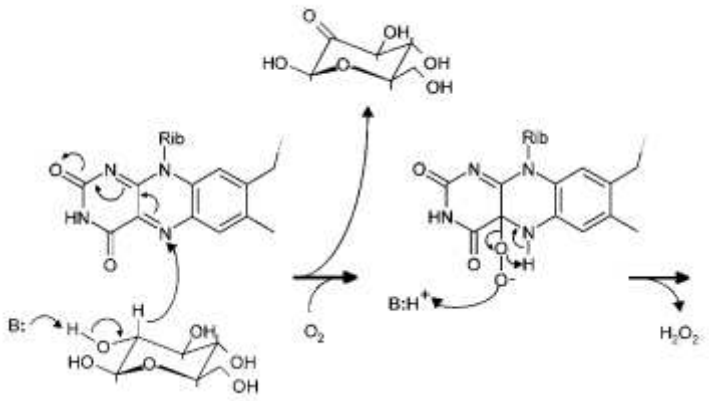

Figura 4 - Rota de ação da enzima POX. Fonte: Retirada do artigo (BANNWARTH et al., 2004).

1.7.2. Família AA4

A família AA4 é formada exclusivamente pela vanilil-álcool oxidases(VAO) (EC 1.1.3.38). Essa enzima é muito estudada por atuar em um amplo espectro de substratos, inclusive os oriundos da degradação da lignina (LEVASSEUR *et al.*, 2013) (Figura 5). A VAO é uma flavoenzima que tem a capacidade de auto catalisar a ligação da flavina na sua estrutura. O trabalho de Fraaije*et al.* (2003), que realizou a mutação sítio dirigida de uma enzima VAO, mostrou que a enzima que perde essa capacidade tem sua taxa enzimática diminuída em até 10 vezes em relação ao tipo selvagem (FRAAIJE *et al.*, 2003). A VAO tem capacidade de catalisar reações de oxidação, desaminação, desmetilação, desidrogenação e hidroxilação, mas tem uma velocidade de reação muito alta para 4-(metoximetil) fenol e álcool vanilil, utilizando-se desses substratos e oxigênio molecular para gerar peróxido de hidrogênio. Devido sua atividade sobre as moléculas oriundas da degradação da lignina e sua alta produção de peróxido de hidrogênio, a VAO é considerada importante em auxiliar o processo de degradação da lignina (GOSWAMI *et al.*, 2013).

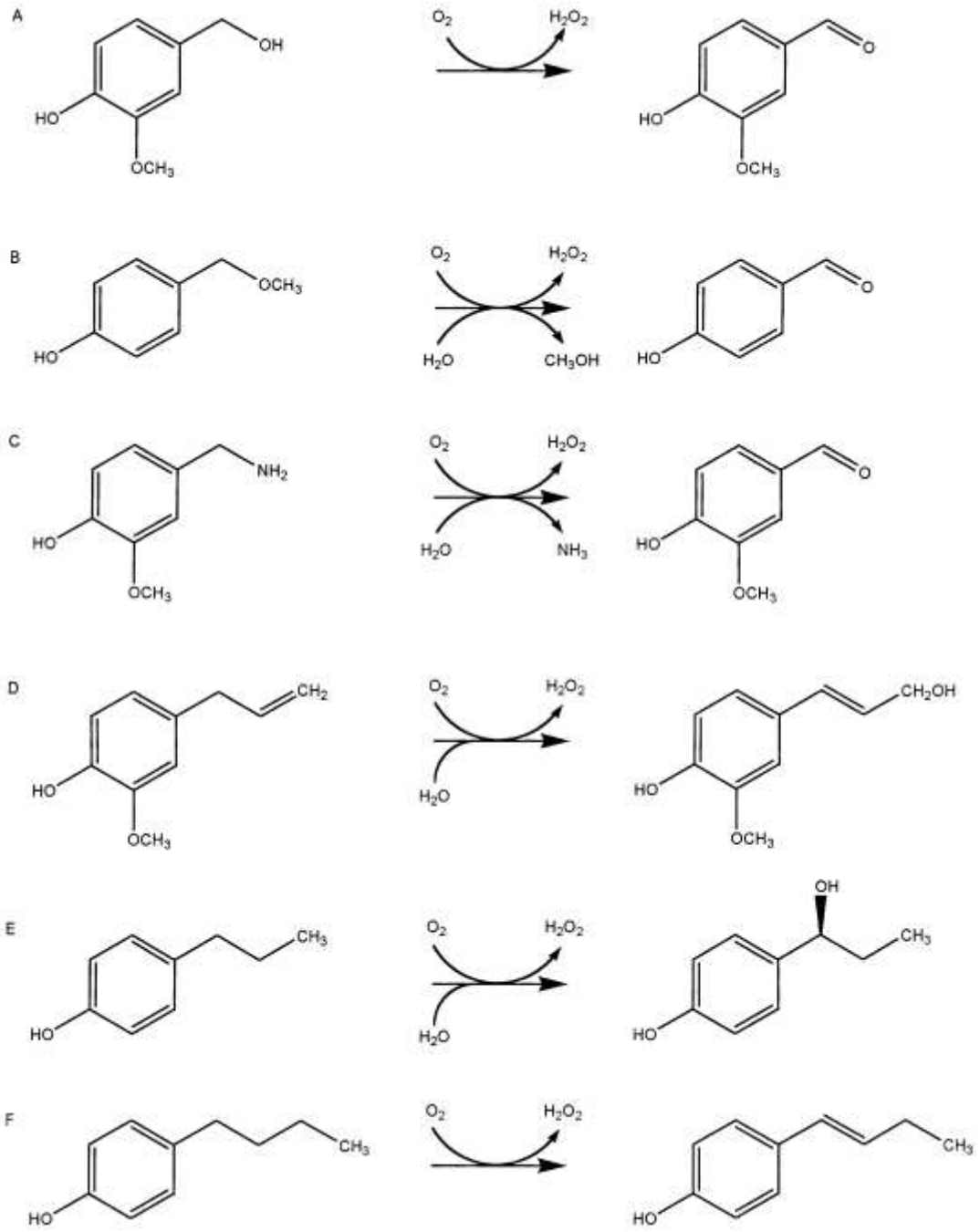

Figura 5 - Reações catalisadas pela enzima VAO, sendo reação A) Oxidação de álcool vanilílico; B) desmetilação de 4-(metoximetil) fenol; C) desaminação de vanililamina; D) hidroxilação de eugenol; E) hidroxilação de 4-propilfenol e F) desidrogenação de 4-butilfenol. Fonte: Retirado de (FRAAIJE et al., 2003).

1.7.3. Família AA5

A família AA5 é composta por oxidase de cobre radical e é dividida em duas subfamílias: AA5sub1 glioxal oxidases (GLOX) e AA5sub2 galactose oxidases (GAO) e álcool oxidase (AO). Essas enzimas têm atividades catalíticas em carboidratos, diferenciam-se

pouco em sua estrutura tridimensional e de sequência, mas sim nos substratos em que atuam sendo a glioxal oxidases atuando em aldeídos diversos, a galactose oxidases em D-isômeros como a D-galactose e a álcool oxidase em álcool alifático (LEVASSEUR *et al.*, 2013).

1.7.3.1. Glioxal oxidases (EC 1.2.3.15)

A glioxal oxidases (GLOX) é uma enzima que contém cobre em sua estrutura, largamente distribuída no reino Fungi, estando presente em fungos de podridão branca, patógenos e fungos simbióticos, sendo uma enzima isolada também em plantas e é descrita em animais e bactérias. A GLOX é muito similar a GAO em análises espectroscópicas, sendo diferente na estabilidade química, potencial redox e propriedades de catálise (DAOU; FAULDS, 2017). A GLOX é capaz de oxidar uma vasta gama de compostos aldeídos, α-hidroxicarbonilo e α-dicarbonilo. Para isso, a enzima consegue utilizar oxigênio molecular gerando como subproduto ácidos orgânicos e peróxido de hidrogênio, tendo como principais substratos metilglioxal e glioxal (KERSTEN*et al.*, 2014). Além da produção de peróxido de hidrogênio, a GLOX produz ácidos orgânicos que podem auxiliar como quelante do Mn^{3+}, melhorando a atividade da MnP. Foi visto também a presença da GLOX junto da MnP nos fluidos extracelulares do basidiomiceto *Ceriporiopsissubvermispora* durante a degradação da lignina. Esses dados ajudam a reforçar a importância da GLOX na degradação da lignina (URZÚA*et al.*, 1998).

1.7.3.2. Galactose oxidases (EC 1.1.3.9) e álcool oxidase (EC 1.1.3.13)

A galactose oxidases (GAO) é uma enzima extracelular que contém um átomo de cobre em sua estrutura que auxilia na reação catalítica. Ela é capaz de oxidar de forma estereoespecífica os D-isômeros de uma larga variedade de álcoois primários, tais como D-galactose, bem como polissacáridos com D-galactose na sua extremidade redutora (PARIKKA*et al.*, 2015; TKAC *et al.*, 2002). A GAO utiliza oxigênio molecular para oxidar os álcoois primários, resultando como subprodutos o peróxido de hidrogênio e seus respectivos aldeídos (DEACON *et al.*, 2004) (Figura 6). A indústria tem grande interesse na utilização da GAO para a produção de materiais de partida para substâncias com potencial comercial elevado, como açúcares Deoxy e N-acetilactosamina. Ela também tem sido estudada para o uso em sensores de açúcar na indústria de alimentos, como a do leite para identificação no nível de lactose (KANYONG *et al.*, 2017).

Estudos mostram atividade da GAO em alguns álcoois como álcool benzílico, 1,2-di-hidroxibutano, mas nenhuma atividade em álcoois alifáticos 1,4-Butanodiol e 4-penteno-1-ol

(SIEBUM *et al.*, 2006). Em estudos recentes, identificaram-se enzimas com estrutura semelhantes a GAO capazes de oxidar eficientemente álcoois alifáticos e com baixa eficiência para galactose e galactosídeos. Essas enzimas foram caracterizadas como álcool oxidases dependentes de cobre, diferenciando das AOX da família AA3 que são dependentes de dinucleótido de flavina e adenina (FAD). Como os demais membros da família AA5, essas enzimas são capazes de utilizar oxigênio molecular para oxidar os álcoois gerando como subproduto aldeídos e peróxido de hidrogênio. Devido a capacidade de produção de peróxido de hidrogênio acredita-se que essas enzimas tenham capacidade de auxiliar as peroxidases na degradação da lignina (YIN *et al.*, 2015).

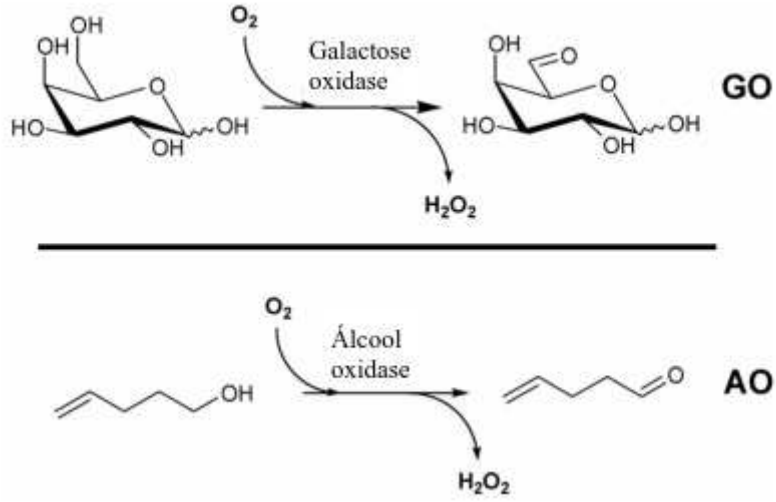

Figura 6 - Sistema de catálise da das enzimas da família AA5sub2. Fonte: Obtidas do artigo (SIEBUM et al., 2006).A figura acima ilustra o princípio de ação das enzimas GAO e AO pertencentes a família AA5sub2. Elas utilizam o substrato e oxigênio molecular para gerar peróxido de hidrogênio.

1.7.4. Família AA6

Essa família consiste pela 1,4-Benzoquinone redutase (BQR) (EC. 1.6.5.6), apresentando a capacidade de reduzir quinonas para hidroquinonas utilizando para isso dinucleótido de nicotinamida e adenina (NADH). As quinonas podem reduzir um ou dois elétrons transformando-se em semi-quinonas ou hidroquinonas (Figura 7). As BQR garantem a redução das quinonas para sua forma mais reduzida, evitando a formação de radicais livres que causam estresse oxidativo nas células (DELLER*et al.*, 2008; KOCH *et al.*, 2017) .

As enzimas da família AA6 também podem estar associadas a capacidade melhorada dos fungos de metabolizar subprodutos da degradação da lignina. Foi visto uma correlação de expressão da BQR com a enzima homogentisato 1,2-dioxigenase, enzima responsável da degradação de compostos aromáticos, quando o fungo *Phanerochaetechrysosporium* é

exposto a compostos como vanilina, indicando que a enzima homogentisate 1,2-dioxigenase apresenta função durante a degradação da lignina (MORI *et al.*, 2016; SHIMIZU *et al.*, 2005). O trabalho de Mori *et al.* (2016) também mostrou que essas enzimas co-expressas em um organismo transformado não altera a atividade na degradação da lignina, mas melhorava a degradação de vanilina podendo-se hipotetizar que a BQR tenham função não apenas de proteção, mas na degradação de compostos aromáticos da lignina (MORI *et al.*, 2016).

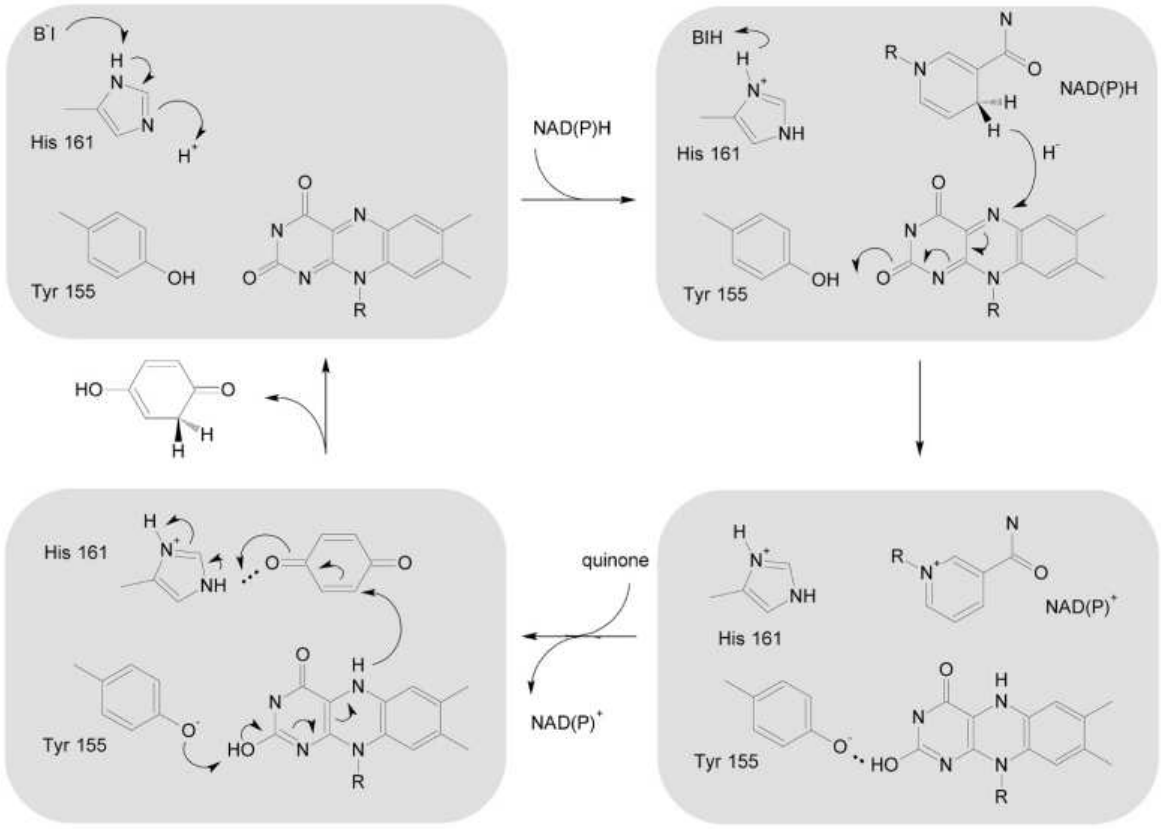

Figura 7 - Mecanismo de reação da enzima BQR. Fonte: Retirado de (DELLER et al., 2008).

1.7.5. Família AA7

Essa família consiste das enzimas glucooligossacarídeo oxidase (GOOX) (EC 1.1.3.-) e quitooligossacarídeo oxidase (CHITO) (EC 1.1.3.-). A GOOX tem capacidade de oxidar diversos tipos de carboidratos, diferenciando-se das demais enzimas pela sua capacidade de oxidar tanto mono, di e oligossacarídeos (FOUMANI*et al.*, 2011; LIN *et al.*, 1991). A CHITO tem atividade em mono e oligossacarídeos N-acetilados, sendo chitotetraose o melhor substrato. Porém ela pode modificar outros carboidratos como glucose, celobiose, mas com menor eficiência (LEFERINK *et al.*, 2008). As duas enzimas (GOOX e CHITO) contêm uma molécula de FAD ligada covalentemente, sendo essencial para o funcionamento da enzima. As enzimas utilizamoxigênio molecular para realizar a oxidação dos respectivos açúcares no

carbono C1, gerando como subprodutos as respectivas lactonas e peróxido de hidrogênio (HUANG *et al.*, 2005) (Figura 8).

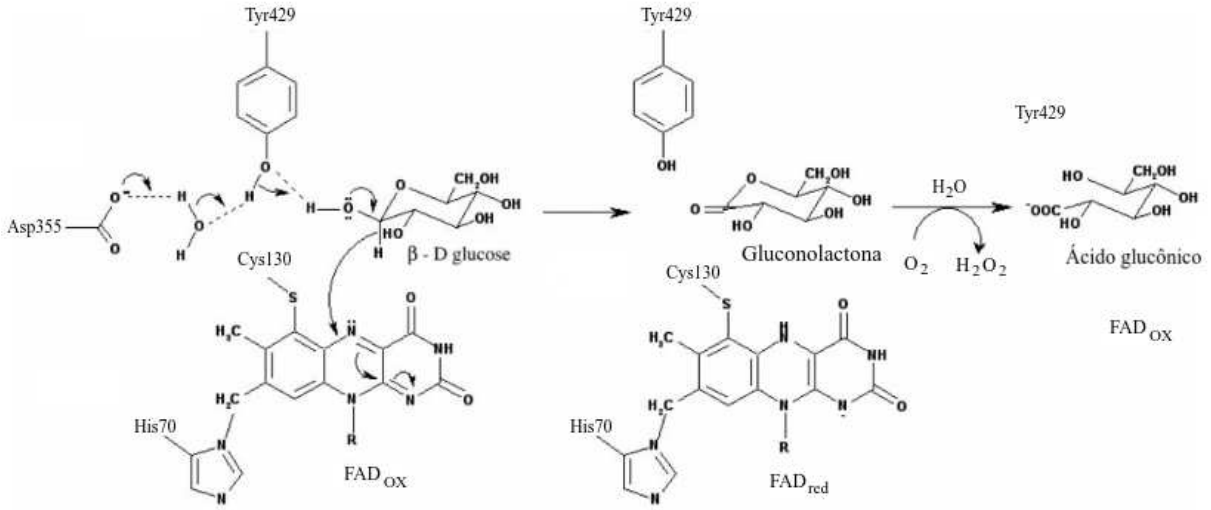

Figura 8 - Mecanismo reacional da enzima GOOX. Fonte: Retirado e adaptado de (HUANG et al., 2005).

1.7.6. Família AA8

A família AA8 consiste no domínio de redutase de ferro (DRF) que é um domínio do citocromo (protoheme IX) da classe espectral b. Ela está presente na estrutura da CDH, podendo ser encontrada isoladamente na natureza (YOSHIDA *et al.*, 2005) . Essa estrutura permite a redução de um elétron como ferricianeto, radicais fenoxi e citocromo c, fazendo com que a hemo-flavoenzima tenha reações na CDH sejam mais rápidas em relação a estrutura sem o domínio DRF (HALLBERG *et al.*, 2000; HENRIKSSON*et al.*, 2000). A DRF pode utilizar o Fe (III) durante o ciclo catalítico, reduzindo-o a Fe (II). O domínio flavina derivado da proteólise de CDH tem a capacidade de reduzir a celobiose na presença de quinonas, mas ele é diminuído para utilizar compostos Fe (III) como receptor de elétrons (CAMERON*et al.*, 2000). Isso sugere que a DRFseja um domínio importante, influenciando na produção de Fe (II), e permitindo a reação de fenton e auxiliando no ataque e degradação da lignina (HENRIKSSON*et al.*, 1993).

1.7.7. Famílias Polissacarídeos lítico monooxigenases

Polissacarídeos lítico monooxigenases (LPMO) (EC 1.14.99.-) são enzimas que utilizam oxidação dependente de cobre, capazes de quebrar ligações glicosídicas de substratos polissacarídicos recalcitrantes como celulose e quitina. As LPMO são divididas no banco de dados CAZy em diferentes famílias: AA9, AA10, AA11, AA13, AA14, AA15 e AA16. Essa separação ocorre não necessariamente devido a função e sim por semelhança de sequência e

tipo regiosseletividade de clivagem (C1, C4 ou outra). Dessa forma podem ocorrer atividades catalíticas semelhantes, como, por exemplo, as famílias AA10, AA11 e AA15 que tem atividade em quitina, podendo apresentar baixa similaridade de sequência, ou mesma conformação de dobra e arquitetura dosítio ativo (CHYLENSKI *et al.*, 2019; FOWLER, 2018; JOHANSEN, 2016).

As LPMOs são ativas em diversos substratos polissacarídicos, que inicialmente foram associadas a atividades enzimáticas em quitina e celulose, mas também foram encontradas ativas em outros substratos como cello-oligosaccharides e várias hemiceluloses. A reação de catálise das LPMOs foi identificada utilizando oxigênio molecular, provocando a oxidação do substrato e produzindo água como subproduto. Posteriormente foi identificado a capacidade do peróxido de hidrogênio ser utilizado como co-substrato, apresentando taxas de reação maiores comparadas com as do oxigênio molecular (CHYLENSKI *et al.*, 2019; HEDEGÅRD; RYDE, 2018; JOHANSEN, 2016). Para a enzima recuperar a atividade, necessita-se de um agente doador de elétrons, capaz de reduzir o cofator de cobre divalente. Para isso, as LPMOs conseguem usar uma grande quantidade de moléculas como: ácido ascórbico, cisteína, flavonóides derivados de plantas, ou compostos fenólicos fúngicos e moléculas formadoras de lignina (FROMMHAGEN *et al.*, 2016; KRACHER *et al.*, 2016).

As LPMOs, quando não estão interagindo como o substrato, são capazes de gerar peróxido de hidrogênio, principalmente no início do ataque da biomassa, momento que os sacarídeos estão protegidos pela lignina,. Para isso, necessitam de oxigênio molecular e de um substrato redutor, sendo capaz de reduzi-lo, formando como subproduto o peróxido de hidrogênio (Figura 9). Na presença de substrato, as LPMOs que não estão interagindo com o substrato podem produzir peróxido de hidrogênio podendo ser posteriormente utilizado pelas LPMOs ligadas na realização da catálise. Essa descoberta é interessante, visto que permitiria a produção de peróxido de hidrogênio por LPMOs, auxiliando as reações das peroxidases principalmente no início do ataque à lignina, momento que a atividade das outras enzimas auxiliares estariam com baixa disponibilidade de substrato para suas reações (BISSARO *et al.*, 2018; LI *et al.*, 2019).

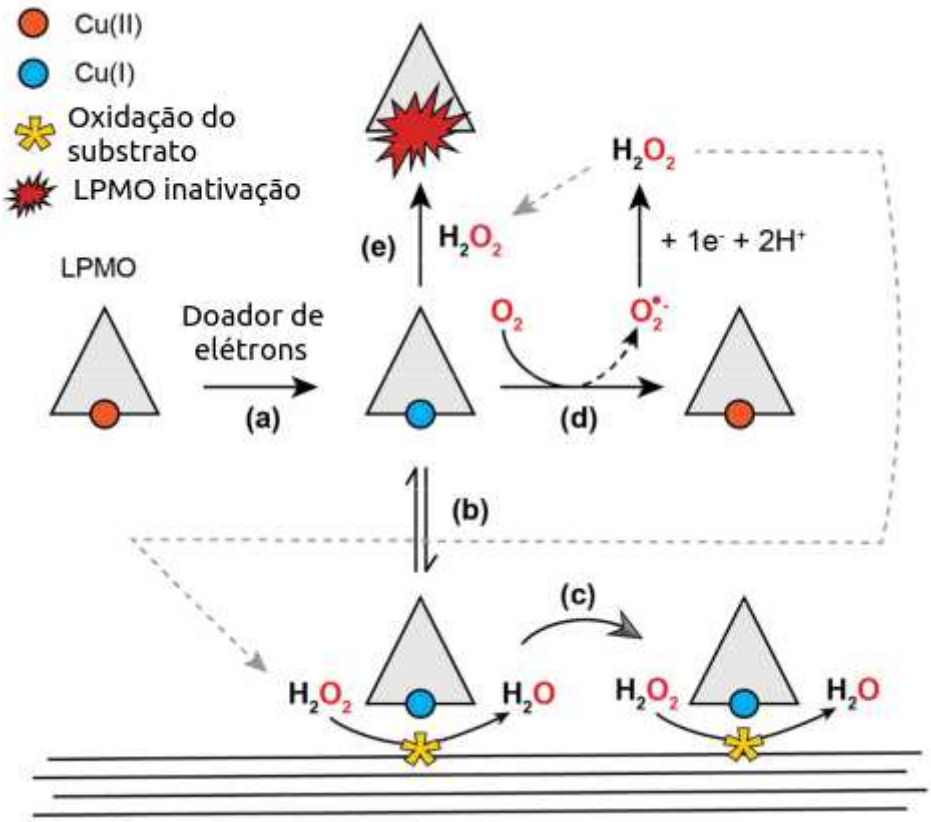

Figura 9 - A ilustração mostra as diferentes reações e situações que a LPMO pode atuar. No estado inicial (a) a enzima com o Cu II é reduzido para Cu I e recupera a atividade enzimática; no estado (b) a enzima pode se ligar ao substrato e utilizar peróxido de hidrogênio em sua reação; no estado (c) a enzima pode continuar a sua reação enquanto for mantido o estado de oxidação do Cu; a enzima livre pode utilizar oxigênio molecular para produzir peróxido de hidrogênio (d) necessitando de um doador de elétrons para recuperar a atividade; no estado (f) se a enzima reduzida reagir com peróxido de hidrogênio pode ocorrer oxidação do sítio ativo levando a inativação da enzima. Fonte: A imagem foi adaptada de (LOOSE et al., 2018).

1.7.8. Família AA12

A família AA12 consiste de enzimas OxirredutasePirroloquinolinaQuinona-Dependente. A primeira enzima encontrada foi uma Piranose desidrogenase (PDH) e foi identificada no basidiomiceto *Coprinopsiscinerea*. A PirroloquinolinaQuinona (PQQ) é um cofator inicialmente descoberto em enzimas bacterianas. A primeira enzima encontrada assemelha-se com as enzimas CDH, pois contém um domínio citocromo AA8, um CBD, mas diferencia-se por ter um domínio de ligação da PQQ ao invés de um domínio de ligação do FAD (MATSUMURA *et al.*, 2014; TAKEDA *et al.*, 2015). A PDH apresenta atividade oxidativa em uma ampla gama de substratos carboidratídicos como L-fucose e uma gama de açúcares raros como: D-arabinose, L-galactose, D-talose e L-gulose (TAKEDA *et al.*, 2019).

1.8. Seleção positiva

A alteração gênica gera o polimorfismo proteico (DESAI *et al.*, 2007). Essas alterações ocorrem por mutações aleatórias no DNA, que são normalmente neutras ou

negativas. Uma mutação é neutra quando a alteração não acarretar em vantagem ou desvantagem adaptativa, dessa forma não modifica a sobrevivência ao organismo portador da variante. A mutação é negativa quando a alteração acarreta em uma desvantagem em questão de sobrevivência para o organismo portador da variante sendo retirada da população durante a seleção natural. A mutação também pode ou não acarretar em melhoria de função ou vantagem adaptativa, aumentando as chances da fixação da variação na população ao sofrer seleção natural e dessa forma ser positivamente selecionada (CHARLESWORTH *et al.*, 2018; DESAI *et al.*, 2007; ELLEGREN *et al.*, 2016). Uma mudança sofre pressão positiva de seleção quando o alelo anterior passa a ser o subótimo, pois o novo alelo é favorecido pela alteração, ocupando a posição de vantagem e isso garante o lugar da nova variação. Uma mutação também pode prosseguir na população se a nova variação causar o mesmo efeito do alelo anterior. Essa variação pode ser adequada a posição, sem garantir um ganho substancial de função ou mesmo desfavorecer a atividade antes realizada, não apresentando caráter adaptativo, levando apenas a geração de polimorfismo (BAZYKIN, 2015).

Se avalia a seleção positiva considerando as substituição não-sinônima maior do que a substituição sinônima. O perfil de substituições não-sinônimas em seleções positivas se deve ao fato de que essas substituições causam alterações nos aminoácidos, provocando uma alteração na estrutura da enzima acarretando em modificações mais drásticas em relação a modificações neutra, sendo as alterações vantajosas e neutras selecionadas durante a seleção natural e as desvantajosas tendendo a ser eliminadas da população (ENDO *et al.*, 1996). Dessa forma durante a identificação de seleção positiva, os efeitos da evolução não-adaptativa podem ser reduzidos, levando-se em consideração o padrão de divergência, pois mutações não-sinônimas têm maior probabilidade de acarretar em melhoria adaptativa comparada a substituições sinônimas. Se a busca de seleção positiva for feita analisando-se alterações não-sinônimas, pode-se garantir maior confiabilidade dos resultados, tendo em vista que os dados de divergência não-sinônima têm relação com a substituição adaptativa, estando menos presente em mutações neutras. Assim, verificando-se substituição não-sinônima na busca de seleção positiva, pode-se aumentar a probabilidade de se encontrar uma alteração adaptativa vantajosa para o organismo (MACPHERSON *et al.*, 2007).

A avaliação é realizada basicamente considerando a razão entre a substituição não-sinônima pela substituição sinônima, possibilitando uma forma direta de mensurar a pressão de seleção em cada códon. As regiões consideradas selecionadas positivamente apresentam razão maior que 1 (AGUILETA *et al.*, 2009). Vale ressaltar que todo resultado encontrado

pela análise da substituição não-sinônima é passível de dúvida, pois nem toda modificação não-sinônima necessariamente resulta em vantagem para o organismo. É necessário entender o grau de pressão que um gene é submetido, caso um gene tenha baixa necessidade para a sobrevivência do organismo ele apresentará maior número de mutações não-sinônimas que poderiam ser aceitas durante o crescimento populacional, gerando para esse gene não-essencial uma falsa impressão de forte seleção positiva (RADMAN*et al.*, 2000). Algo semelhante pode ser encontrado em genes duplicados, podendo um gene manter a função, permitindo que o gene duplicado venha a sofrer maior grau de mutação não-sinônima, caso esse gene que sofreu mais mutação fosse usado para identificação de seleção positiva poderia gerar a falsa impressão de seleção. Dessa forma, alguns eventos podem acarretar em uma falsa identificação de evento de pressão positiva de seleção, fazendo com que eventos identificados como seleção positiva, utilizando-se de substituições não-sinônimas, necessitem de confirmação experimental para comprovar o ganho de vantagem adaptativa (FORCE *et al.*, 1999). Mesmo assim a análise de seleção positiva ajudou e ainda ajuda na identificação de genes que são funcionalmente importantes para os organismos, sendo que a melhoria das suas funções acarretam em vantagens de crescimento e sobrevivência. Dessa forma a seleção positiva pode ajudar a identificar mecanismos enzimáticos importantes para a sobrevivência de um organismo (BISWAS*et al.*, 2006).

2. OBJETIVOS

2.1. Objetivo Geral

Utilizar os metadados e as sequências contidas no banco CAZy relacionados às enzimas fúngicas pouco estudadas associadas a degradação da lignina das categorias AA3 até a AA16, permitindo obter os dados taxonômicos correlacionados à cada uma delas

2.1.1. Objetivos Específicos

- Construir um banco de dados de sequências com os respectivos metadados taxonômico e existentes no banco CAZy, para cada enzima fúngica pouco estudada associada à degradação da lignina, das categorias AA3 até a AA16 contido no banco CAZy;
- Analisar a distribuição taxonômica, distribuição de tamanho de sequência e isoformas existentes;
- Alinhar as sequências e identificar as domínios conservadas e variáveis (padrões estruturais) a fim de relacionar os domínios identificados dentro dos alinhamentos (para cada uma das famílias de enzimas individualmente);
- Identificar regiões que estejam sob seleção positiva;
- Modelagem das sequências com evidência de seleção positiva e analisar a região selecionada positivamente a fim de aumentar a confiabilidade dos resultados com dados da literatura sobre efeitos de mutações na estrutura tridimensional.

3. JUSTIFICATIVA

A biomassa vegetal é uma alternativa viável e sustentável para substituir os combustíveis fósseis, porém sua utilização como recurso renovável é limitada pela dificuldade do seu processamento. A lignina existente no material lignocelulósico protege as cadeias polissacarídicas da degradação e disponibilização dos açúcares para fermentação, necessitando de um pré-tratamento para aumentar o rendimento dos açúcares fermentáveis. Os processos físicos e químicos de pré-tratamento da biomassa apresentam desvantagens que tornam esses processos financeiramente inviáveis. Uma alternativa seria os processos de pré-tratamento biológicos, pois apresentam grandes vantagens em comparação aos demais processos, mas ainda apresentam fatores que inviabilizam sua utilização. Dessa forma desenvolver processos biológicos mais eficientes que reduza as suas desvantagens torna-se uma forma de baratear a produção de biocombustíveis e permitiria a construção de um mundo mais verde.

Se sabe que uma das dificuldades da utilização das enzimas lignolíticas é a necessidade de moléculas mediadoras para uma degradação eficiente, além disso as peroxidases necessitam de peróxido de hidrogênio para conseguirem atuar. Uma alternativa para baratear o processo é a utilização de enzimas produtoras de peróxido de hidrogênio, porém as enzimas produtoras de peróxido de hidrogênio não são bem estudadas e ainda falta muitas informações básicas para a sua utilização. Dessa forma, buscando expandir o conhecimento sobre elas desenvolvemos nossos trabalhos em cima das enzimas fúngicas pouco estudada associadas a degradação da lignina das categorias AA3 até a AA16 do banco CAZy, analisando os domínios nelas encontrados, além de utilizar os dados de domínios obtidos para melhorar a curadoria das sequências presentes no banco CAZy. Paralelo a isso utilizamos a análise de seleção positiva para identificar alguma evidência de pressão nesses grupos de enzimas e utilizando a modelagem molecular para analisar os locais em que essas seleções se encontram. Aproveitamos também para entender a distribuição taxonômica dentre as enzimas existentes no CAZy, verificar a existência de isoformas e gerar histogramas da distribuição do tamanho dessas sequências, permitindo gerar informação de pesquisa básica sobre os dados existentes.

4. MATERIAIS E MÉTODOS

A construção do banco de dado e as análises realizadas foram feitas em várias etapas. No primeiro momento foi realizado o download e construção do banco de dados contendo as informações do CAZy, download das sequências e download dos dados taxonômicos. Vale lembrar que o CAZy apresenta os dados de cada família AA divididas entre os domínios taxonômicos, sendo mais fácil baixar todos os códigos de depósitos de eucariotos no CAZy e posteriormente selecionar as sequências fúngicas.

Após o banco construído, foi realizado a retirada de sequências com múltiplas cópias. Posteriormente, foi averiguado alguns dados faltantes que não estavam ranqueados. Por fim, foram retiradas sequências não-fúngicas e feito a finalização da construção do banco para o trabalho.

As análises realizadas foram: (i) construção dos gráficos de isoformasprotéicas, (ii) de distribuição taxoôomica e (iii) tamanho de sequências. Também foi feito o (iv) alinhamento, (v) análise dos domínios, (vi) análise de regiões sob provável seleção positiva e (vii) modelagem das sequências com evidência de seleção positiva para averiguar a localização espacial do sítio selecionado (Figura 10).

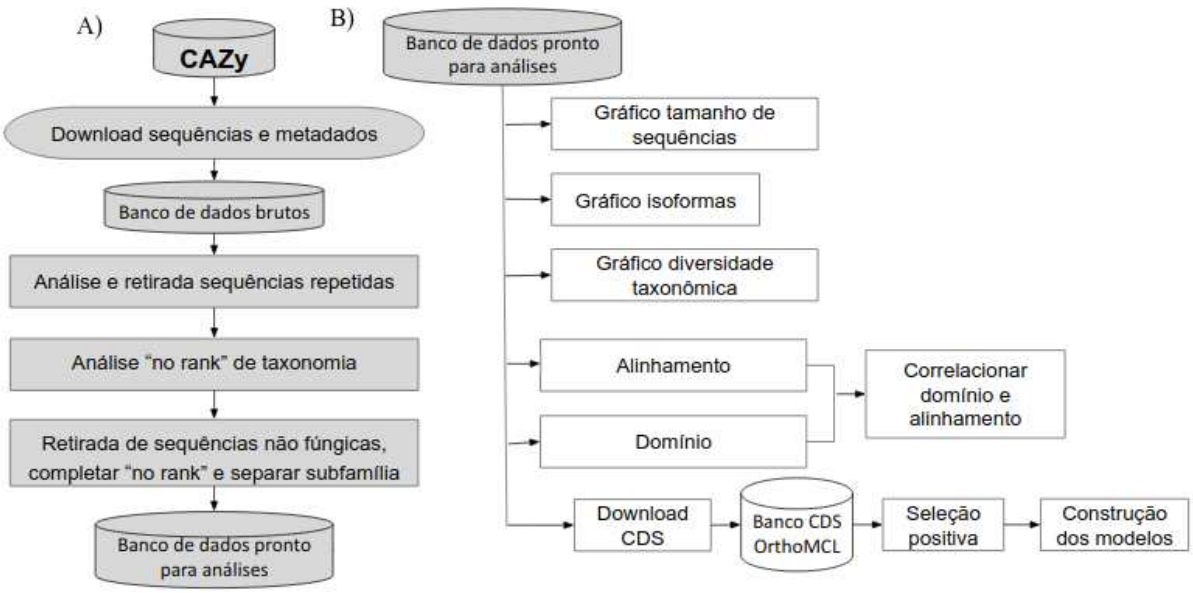

Figura 10 - Fluxograma geral do trabalho. A) Construção dos banco de dados; B) Análises realizadas com os bancos de dados após curadoria.Cada caixa representa uma etapa do processo. O banco de dados brutos representa o banco de dados CAZy. As caixas em cinza são etapas de filtragem e as em branco são etapas de processamento de dados.

4.1. Obtenção das informações e geração do banco de dados

O processo de obtenção dos dados dividiu-se em três etapas. A primeira etapa constituiu-se pela construção do início das tabelas de metadados com as informações contidas no CAZy. Vale lembrar que os dados baixados nessa etapa consistiram de informações sobre enzimas eucarióticas contidas nas famílias AA3 até AA16, pois o CAZy dividia as informações dentro de cada família AA por domínios taxonômicos, sendo mais fácil baixar todas as sequências eucarióticas e separar posteriormente. No primeiro momento foi obtido as informações tabeladas pelo CAZy:

- Nome da proteína: O nome da proteína consiste do nome utilizado pelo banco de dados CAZy para cada proteína no banco;
- Classificação EC: A classificação *EnzymeCommissionNumbers* (EC) é a classificação da enzima baseada na reação que catalisa. Vale ressaltar, no entanto,que a maioria das enzimas não receberam uma classificação EC;
- Organismo onde foi encontrada a enzima;
- ID: O ID (identificação) consiste na identificação utilizada nos bancos de dados em que as sequências dessas enzimas se encontram, podendo estar contidos no GenPept, Uniprot e *Protein Data Bank* (PDB);
- Subfamília: Visto que algumas famílias AA continham subgrupos, essas informações também foram acrescentadas à tabela;
- Site de taxonomia: O endereço web de taxonomia indicado pelo CAZy. Esses endereços web também foram acrescentados na tabela para posterior *download* do código-fonte e extração dos dados de taxonomia.

A segunda etapa compreendeu a obtenção das sequências de aminoácidos nos bancos de dados onde estavam armazenadas, utilizando, para isso, os IDs de depósito obtidos no CAZy. As sequências estavam armazenadas, principalmente no GenPept e no PDB. A terceira etapa consistiu no *download* do código-fonte do endereço web, obtido durante a primeira etapa, presentes nos arquivos eukaryota_taxonomia.txt contendo a classificação taxonômica dos organismos. Os endereços web obtido durante a primeira etapa, presentes nos arquivos eukaryota_taxonomia.txt eram referentes a base de dados de taxonomia do *National Center for BiotechnologyInformation* (NCBI) (AGARWALA *et al.*, 2018). Vale ressaltar que foi obtido os dados taxonômicos para cada sequência baixada. Os dados foram obtidos entre o período do dia 22 a 25 de março de 2019. Os dados trabalhados foram referentes às categoriais *AuxiliaryActivity*(AA), da família AA3 até a AA16 (Figura 11).

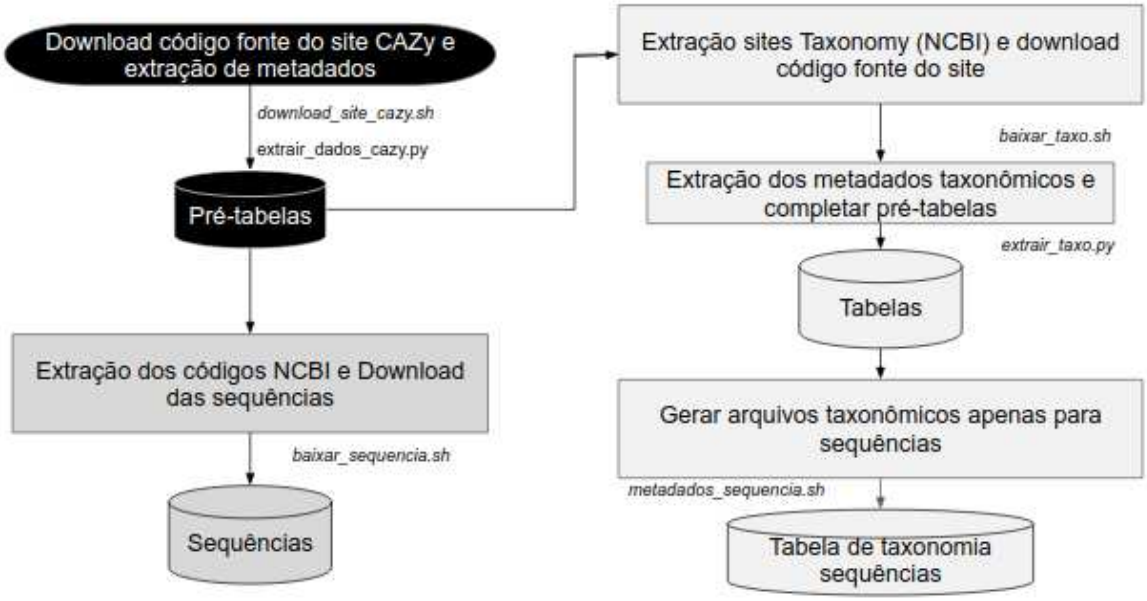

Figura 11- Fluxograma de download do banco original. Os nomes dos scripts autorais desenvolvidos em BASH ou Python para realizar cada tarefa estão indicados em itálico. O fluxograma demonstra como ocorreu a construção do banco de dados. As caixas de cor preta representam a primeira etapa de processamento, as caixas de cor cinza, a segunda etapa e as caixas em branco, a terceira etapa. Os textos ao lado das caixas foram os scripts utilizados para realizar o processo.

Para realizar a primeira etapa, foi feito o *download* do código-fonte das páginas do site do CAZy para cada família AA que seria analisada, que foi processado utilizando padrões de escrita do código-fonte do endereço web para obter o nome das sequências e seus respectivos metadados. Para tal, foram escritos pelo autor do trabalho e utilizados dois *scripts*, *download_site_cazy.sh* e *extrair_dados_cazy.py* (Apêndice A). Os dados extraídos foram organizados em pré-tabelas (existe uma para cada família AA), às quais foram acrescentadas as informações de taxonomia obtidos na terceira etapa. Os arquivos foram nomeados como eukaryota_taxonomia.txt (https://drive.google.com/drive/folders/1Ym7jc6Xwl7C-ehGA1UXtXZhKXjTjSYUM?usp=sharing).

Na segunda etapa foi realizado o *download* das sequências de aminoácidos da base de dados GenPept (NCBI) a partir dos IDs nos acessos do CAZy contidos nos arquivos eukaryota_taxonomia.txt, usando o *script* autoral *baixar_sequencia.sh* (Apêndice A). As proteínas que apresentavam apenas ID do banco PDB foram baixadas manualmente para cada família AA, pois poucas sequências apresentavam essa característica. As sequências baixadas se encontram nos arquivos eukaryota_sequencia.fa(https://drive.google.com/drive/folders/1Ym7jc6Xwl7C-ehGA1UXtXZhKXjTjSYUM?usp=sharing) e foram organizadas em arquivos FASTA entre as famílias AA do CAZy, gerando um total de 14 arquivos FASTA diferentes,

um para cada família. Algumas enzimas que constavam no banco de dados CAZy não apresentavam nenhum código. Sendo assim, essas enzimas foram desconsideradas na construção do banco de sequências.

Os downloads das sequências foram realizados no dia 22 de março do ano de 2019. Em seguida, foi realizado a verificação dos dados conferindo-se o número e os IDs das sequências baixadas com o número de sequências na tabela da primeira etapa e com os IDs do banco para verificar, respectivamente, se havia alguma não-conformidade.

A terceira etapa consistiu da obtenção da taxonomia de cada sequência, para isso foi utilizado o endereço web de cada identificação taxonômica obtida durante a primeira etapa, presentes nos arquivos eukaryota_taxonomia.txt que estão dentro das pastas (https://drive.google.com/drive/ folders/1Ym7jc6Xwl7C-ehGA1UXtXZhKXjTjSYUM?usp=sharing). Foram realizados os *downloads* e extraídos os dados de taxonomia de cada página baixada, acrescentando as informações nas pré-tabelas de cada família AA.

Os dados taxonômicos que não estavam presentes ou que não foram possíveis de extrair, eram completados com a palavra "vazio". Algumas das enzimas descritas no banco CAZy não apresentaram um ID de armazenamento associado, sendo necessário criar tabelas para cada família AA com os metadados apenas das sequências baixadas na segunda etapa, os quais foram salvos nos arquivos taxonomia_sequencias.txt que estão dentro das pastas (https://drive.google.com/drive/folders/1Ym7jc6Xwl7C-ehGA1UXtXZhKXjTjSYUM?usp=sharing). Nessa etapa o autor escreveu e utilizou os *scripts* *baixar_taxo.sh*, *extrair_taxo.py* e *metadados_sequencia.sh* (Apêndice A).

Foi realizado então a construção do banco de sequências dentro da *AuxiliaryActivities*(AA), para as famílias AA3 até a AA16 do CAZy. Os arquivos utilizados para as etapas de filtragem foram os referentes aos dados de cada família AA, taxonomia_sequencias.txt e eukaryota_sequencia.fa.

4.2. Análise de sequências repetidas

No primeiro filtro foi verificada a existências de sequências repetidas entre as famílias AA. Também foi verificada a existência de sequências repetidas dentro da mesma família. O segundo filtro leva em consideração a espécie ou o gênero associado à sequência. Caso a mesma sequência estivesse presente em organismos de espécies diferentes, ela permanecia no banco de dados, caso tivesse ausência de dados de espécie, era utilizado dados de gênero e caso não houvesse dados de gênero, então as sequências iguais eram descartadas, mantendo-

se apenas uma representante. A escolha por manter sequências iguais oriundas de organismos filogeneticamente distantes foi feita devido a diversidade taxonômica ser algo que será analisado e essa informação pode ser válida para futuros leitores, além disso eventos desse tipo foram raros e mencionados nos resultados. A verificação e a retirada das sequências com múltiplas cópias foram realizadas pelos *scripts* de autoria própria, *analise_seq_duplicada.py* e *remove_seq_duplicada.py* (Apêndice B).

4.3. Análise de erros nas identificações taxonômicas e finalização da construção do banco de sequências

Algumas identificações taxonômicas estavam recebendo classificação genérica pelo Taxonomy (NCBI). Essa classificação apresentava padrão de escrita diferente, apresentando "no rank" no local onde estaria o nível taxonômico, seguido de uma descrição de qual seria a classificação taxonômica. Por exemplo, o correto seria "espécie = *Fusariumannosum*", porém esta recebia ao invés disso "no rank = *Fusariumannosumspeciescomplex*". Com isso, essa classificação impedia que fossem anotadas no banco de dados de forma automática, necessitando de verificação manual. Essa classificação poderia aparecer no meio de dois níveis taxonômicos (ex. família "no rank" espécie) ou como classificação taxonômica mais inclusiva (ex. família gênero "no rank"). Os dados que tinham como nível taxonômico "no rank" eram preenchidos na tabela de taxonomia com a palavra "vazio". Os dados foram então identificados computacionalmente, localizando-se em qual organismo ocorriam os eventos de "no rank". Depois de identificados, cada caso era averiguado manualmente e os dados contidos no "no rank" eram acrescentados nas tabelas de taxonomia manualmente (Figura 12).

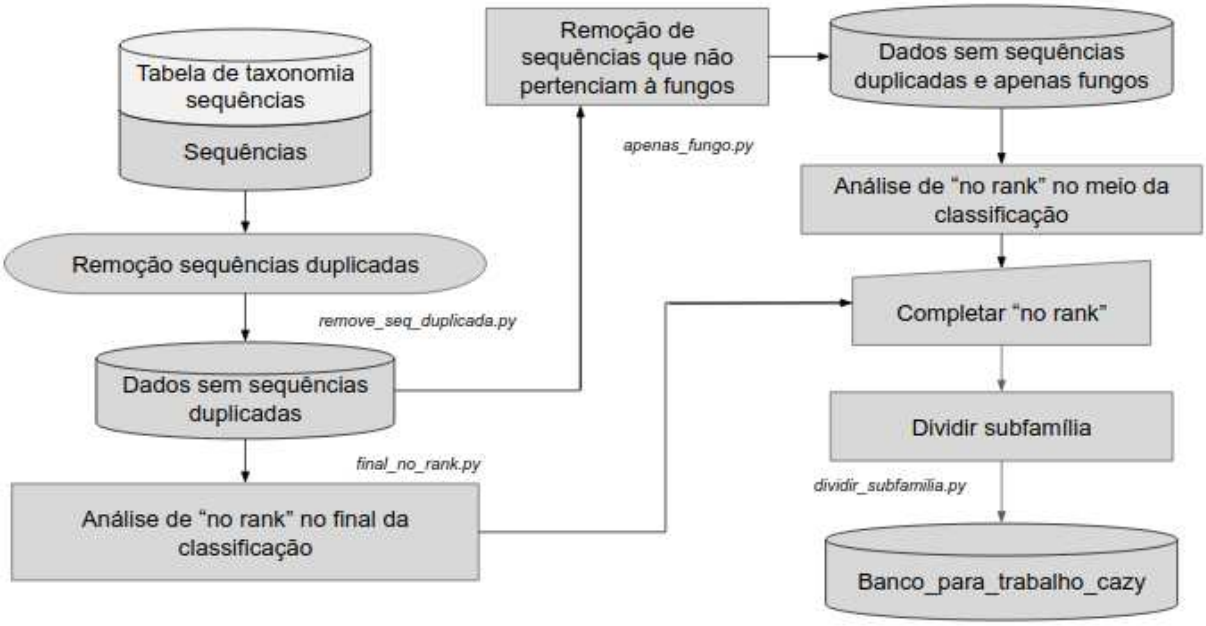

Figura 12 - Curadoria e finalização do banco de dados.O banco de dados de entrada recebera a mesma indicação de cor das etapas de saída da seção 4.1. Aqui se encontram as etapas de curadoria dos dados e finalização do banco de dados. Os textos ao lado das caixas foram os scripts utilizados para realizar o processo.

Cada situação de "no rank" foi analisada separadamente. Para isso, identificaram-se os arquivos dos endereços web obtidos durante a primeira etapa, presentes nos arquivos eukaryota_taxonomia.txt que apresentavam o termo "no rank" como última classificação e as família AA a qual pertenciam esses arquivos, utilizando-se nessa etapa o *script* feito pelo autor *final_no_rank.py*. Para identificar qual organismo tinha "no rank" entre duas classificações taxonômicas utilizaram-se *awk* e *grep* no terminal do Linux; o comando é mostrado no (Apêndice B). Depois de identificados, os organismos com "no rank", o endereço web obtido durante a primeira etapa presente no arquivo (eukaryota_taxonomia.txt) era aberto manualmente e a classificação correta era obtida. Posteriormente, os dados corretos foram acrescentados manualmente nos dados de taxonomia.

Foi observado que todos os fungos recebiam a classificação em reino. Como apenas sequências fúngicas seriam avaliadas, essas sequências foram filtradas antes de acrescentar os dados manualmente, utilizando-se o *script* autoral *apenas_fungo.py* (Apêndice B), dessa forma facilitou-se a correção dos "vazios". Para verificar se nenhum fungo tinha a classificação "vazio" no nível de reino, foram utilizados os comandos *awk* e *grep* no terminal do Linux (Apêndice B), isolando-se as linhas sem informação taxonômica na posição referente a reino e avaliando-se os casos que não apresentavam classificação no nível de reino.

Após a filtragem e análise do banco de dados, foi realizada a separação das subfamílias das famílias AA3 e AA5, utilizando-se o *script* de autoria própria

dividir_subfamilia.py (Apêndice B)e criação de tabelas individuais para cada subfamília. A família AA3 era composta por quatro subfamílias e a AA5 por apenas duas. Algumas sequências não apresentavam subfamílias definidas pelo CAZy, sendo assim não estão presentes em nenhum dos arquivos gerados. As análises posteriores utilizaram os dados das subfamílias individualmente. No final dessa etapa, o banco de dados constituído estava corrigido e pronto para ser utilizado. Os dados finais das sequências foram salvos de acordo com os dados de cada família AA sequencia_final_fungo.fa e os dados de taxonomia finais foram salvos nas tabelas taxo_final_fungo.txt (https://drive.google.com/drive/folders/1Y_Hyv8Nu4o1O3H7X K4xqsrhmzUsBXWAq?usp=sharing). Esses dois arquivos formam o Banco_para_trabalho_cazy.

4.4. Análise das isoformas, distribuição do tamanho das sequências e distribuição taxonômica

Para a obtenção do número de isoformas em cada grupo, utilizou-se o *script* de autoria própria *isoformas.py* (Apêndice C), esse *script* utilizava os dados de taxonomia para verificar quantas sequências diferentes cada espécie tinha, gerando como saída um documento contendo o número de isoformas para cada espécie dentro de cada família AA. Para a análise da diversidade taxonômica, foi utilizado o *script* de autoria própria *tabelando_diversidade.py* (Apêndice C),oqual retorna uma tabela para cada nível taxonômico contendo as identificações taxonômicas existentes e o número de indivíduos. Para obter os tamanhos das sequências dentro de uma família, utilizou-se o *script* de autoria própria *tamanho_sequencias.py* (Apêndice C), o qual retorna uma tabela contendo o ID e o tamanho da sequência. Os dados taxonômicos, isoformas e distribuição de tamanho foram avaliados para cada família individualmente.

Os gráficos de diversidade taxonômica foram criados, usando-se o *scriptgrafico_diversidade.R*(Apêndice C) e o gráfico de isoformas foi construído, utilizando-se o *scriptgrafico_isoforma.R*(Apêndice C) ambos de autoria própria. O gráfico do tamanho das sequências foi construído utilizando-se o *script* de autoria própria*histograma.R*(Apêndice C) (Figura 13). Com esses dados foi possível ter uma ideia do número de isoformas existentes no bancos de dados para cada organismo, visualizar de forma mais fácil quais organismos são mais estudados e gerou uma forma mais prática de verificar quais organismos apresentam uma determinada enzima. O histograma também permitiu gerar informações quanto a

diversidade de tamanho das sequências contidas no banco. Essas informações oriundas de pesquisa básica podem auxiliar futuros leitores com informações não-convencionais.

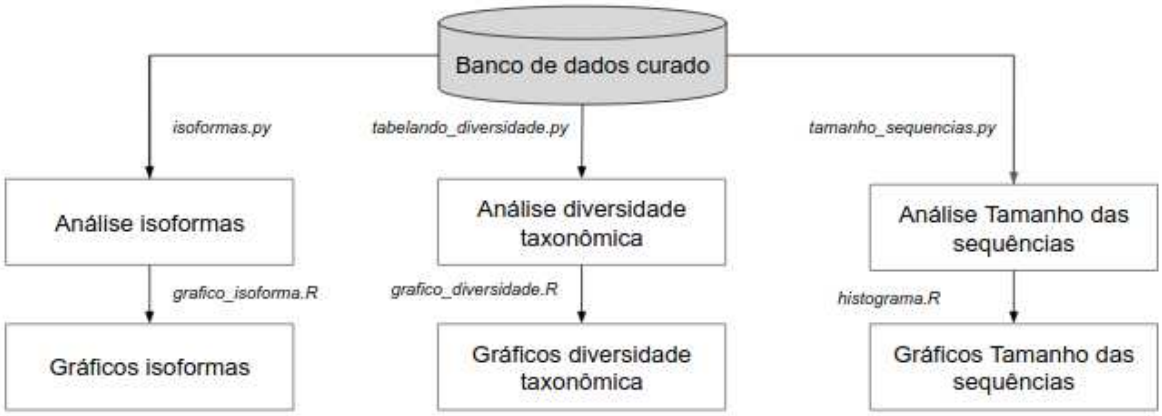

Figura 13 - Fluxograma indicando a análise e construção dos gráficos de isoformas, tamanho das sequências e diversidade taxonômica.O cilindro em cinza representa o banco de dados após filtragem obtidos como produto da seção 4.3, as caixas em branco representam os processos de análise dos dados e construção dos gráficos e os textos ao lado das caixas foram os scripts utilizados para realizar o processo.

4.5. Construção dos alinhamentos e análise dos domínios

O alinhamento das sequências foi feito utilizando-se o programa MAFFT v7.427 (ROZEWICKI *et al.*, 2019) com o parâmetro --localpair (algoritmo mais preciso) e o parâmetro --maxiterate 1000 que define o valor máximo de iteração de refinamento do alinhamento. Os dados foram convertidos para Fasta utilizando-se o *script* autoral *convert_clustal_fasta.py* (Apêndice D) e as análises dos domínios foi realizada utilizando-se o programa Pfam v1.6 (instalado localmente) (EL-GEBALI *et al.*, 2019). Foi instalado também o hmmer v3.2.1 (HMMER, 2019) e o Bioperl v1.6.924, pois o Pfam os utiliza em sua execução (ou seja, são dependências do Pfam)

Os arquivos com os bancos de dados utilizados pelo Pfam foram obtidos no endereço web ftpPfam. Os nomes foram Pfam-A.hmm, com data de atualização 3 de setembro de 2018, Pfam-A.hmm.dat, com data de atualização em 28 agosto de 2018 e active_site.dat, com data de atualização 28 de agosto de 2018. Os parâmetros utilizados foram os mesmos do Pfam*online*. As sequências utilizadas foram as do banco sequencia_final_fungo.fa gerado na saída da seção 4.3. Os alinhamentos (https://drive.google.com/drive/folders/1khK52SqBkaMTjFm_TBkt28ziApWalKECo) e os domínios (https://drive.google.com/drive/folders/1Le5ZSzjsk2-U_BSmTC0lFJC27J1c0qlf8) encontrados estão *online* como alinhamento.

A análise dos domínios foi feita, utilizando-se os autorais *scriptseditar_entrada.py* (Apêndice D) e *analises_dominio.py* (Apêndice D). A análise dos domínios se deu utilizando-se o e-value de $9.9e^{-5}$. A análise dava como saída quatro arquivos: o primeiro arquivo contendo, em forma de matriz, a identificação da sequência e os domínios com E-value satisfatório para cada sequência; o segundo arquivo era a faixa global em que os domínios apareciam no alinhamento; o terceiro arquivo era a faixa no alinhamento individual de cada domínio de cada sequência; e o quarto arquivo mostrava quais sequências não apresentavam um domínio específico e quais apresentavam esse domínios múltiplos (Figura 14). As informações dos domínios existentes e se eles se alinham de forma próxima auxilia em processos como de identificação de possíveis fragmentos e enzimas quiméricas. Como pode ser visto na etapa 4.6 na qual foi realizado a seleção positiva, tais informações auxiliaram na identificação de fragmentos que quando não foram acrescentadas faziam com que esses fragmentos entrassem no processo. A identificação de domínios também auxiliou na identificação de possíveis mecanismos ainda não descritos na literatura que podem auxiliar na degradação da biomassa vegetal. Os resultados das correlações são encontrados *online* na pasta analise_resultados_dominios(https://drive.google.com/drive/folders/1boMp6BJxCFDneH129QP9lEuJEmT5GI3f).

Figura 14 - Fluxograma de construção e análise dos domínios conservados. O cilindro em cinza representa o banco de dados após filtragem obtidos como produto da seção 4.3. As caixas em branco representam os processos de alinhamento de sequência, obtenção dos domínios e correlacionamento entre os dois. Os textos ao lado das caixas foram os scripts utilizados para realizar o processo.

4.6. Análise de Seleção positiva

Para realizar a análise de seleção positiva, foi necessário realizar o *download* das sequências CDS do banco GenBank do NCBI para todas as sequências de aminoácidos referentes ao banco de sequências proteicas sequencia_final_fungo.fa, gerado na saída da seção 4.3. As CDS baixadas foram comparadas com as sequências de aminoácidos correspondentes. Para isso, elas foram traduzidas e conferidas com suas respectivas sequências de aminoácidos, sendo selecionadas apenas as CDS com tradução idêntica as sequências de aminoácidos, utilizando-se os *scripts* autorais *baixar_cds.sh* e *comparar_cds_aa.py*. Por fim, foram copiados os dados taxonômicos e as sequências proteicas do banco Banco_para_trabalho_cazy, gerado na saída da seção 4.3. Para organizar a entrada para o programa POTION v1.0 (HONGO *et al.*, 2015) (que avalia a probabilidade de ocorrência de seleção positiva), foi utilizado o *script* autoral *entrada_ortho.py* (Apêndice E). Ao final, tem-se os dados organizados na entrada OrthoMCL referentes aos dados de cada família AA, dentro da pasta cds, que contém os arquivos cds_ortho_fungo.fa, taxo_ortho_fungo.txt e sequencia_ortho_fungo.fa (https://drive.google.com/drive/folders/1j7zWG_CTyMgMN5bVMcm3-wCRsEVNWh81).

As sequências dentro das famílias AA do CAZy podem variar amplamente de tamanho, sendo difícil identificar possíveis fragmentos. Dessa forma, foram utilizados os dados referentes aos domínios conservados, identificando quais domínios eram importantes para a atividade da enzima dentro de cada grupo AA, filtrando aqueles que não apresentavam o domínio dentro da média de cada grupo AA, com o *script* autoral *filtrar_dominio.py* (Apêndice E). Essa etapa foi importante, pois, antes da sua adição, sequências que não apresentavam domínios e que eram prováveis fragmentos passaram pelo filtro do POTION, podendo atrapalhar a identificação de seleção positiva.

O POTION realiza as análises de seleção positiva, utilizando ortólogos um para um (1:1). Para encontrar as sequências ortólogas, necessitou-se usar o programa OrthoMCL v2.0.9 (ORTHOMCL, 2019). O OrthoMCL necessita de uma etapa de aplicação do algoritmo BLAST *AllvsAll*, no qual foi substituído pelo programa Diamond v0.9.25 (BUCHFINK *et al.*, 2015) devido sua capacidade de processamento superior. Além disso, os arquivos de entrada (sequências) do programa OrthoMCL têm que estar em arquivos individuais referentes a cada espécie. Assim, foi necessário dividir as sequências em arquivos para cada espécie de forma a agrupar as sequências parálogas, utilizando-se os *scripts* autoral *agrupar_por_especie.py* (Apêndice E). Para identificar a qual espécie cada sequência pertencia, utilizaram-se os dados

de taxonomia de cada sequência contidos no arquivo taxo_ortho_fungo.txt (https://drive.google.com/drive/folders/1j7zWG_CTyMgMN5bVMcm3- wCRsEVNWh81).

Como algumas sequências não apresentaram classificação ao nível de espécie, foi utilizada a classificação ao nível de gênero, agrupando-se em um único arquivo as sequências referente ao gênero. Caso esse gênero já existisse em uma sequência classificada ao nível de espécie, as sequências classificadas apenas com gênero eram descartadas. Logo, desconsideravam-se sequências sem classificação de espécie e com classificação de gênero, caso uma sequência com aquele gênero tivesse classificação de espécie dentro da família analisada.

O OrthoMCL aceita arquivos com até três algarismos, sendo necessário renomear os arquivos gerados pelo *script* autoral *agrupar_por_especie.py* que gerava saídas com o nome completo da espécie. Para essa tarefa, utilizou-se o *script* autoral *renomear.sh* (Apêndice E). Além disso, para o uso do programa, foram utilizadas as configurações padrões disponibilizadas pelo OrthoMCL.

O arquivo gerado pelo OrthoMCL precisa ser convertido para um formato de entrada específico do POTION. Para isso, foi utilizado um *script* chamado *convert_orthoMCL_versions.pl*, o qual está presente nos arquivos do POTION. O programa POTION foi executado com as configurações descritas no arquivo conf_potion.conf (Apêndice E).

A análise das seleções positivas foi feita manualmente, comparando-se a região selecionada positivamente com a sequência referência indicada na saída POTION, utilizando-se a função "Localizar" do programa *gedit* para copiar a sequência e contar os caracteres utilizando o comando *wc -c* no terminal Linux. Depois de identificada a posição da seleção positiva, esses dados eram comparados com os dados das sequências do grupo selecionado positivamente, usando-se o *script* autoral *dados_grupo_selecao_positiva.py* (Apêndice E). Os grupos selecionados com seleção positiva podem estar relacionados com vantagens evolutivas que podem ser analisados futuramente utilizando-se de trabalhos de bancada. O *script dados_grupo_selecao_positiva.py* obtinha as informações de domínio, posição dos domínios no alinhamentos e tamanho do domínio na sequência, utilizando, para isso, os dados da etapa 4.5. Essa análise permite verificar se uma seleção estava dentro de uma região de domínio e localizar a seleção dentro do alinhamento e posteriormente localizar no modelo tridimensional predito (Figura 15).

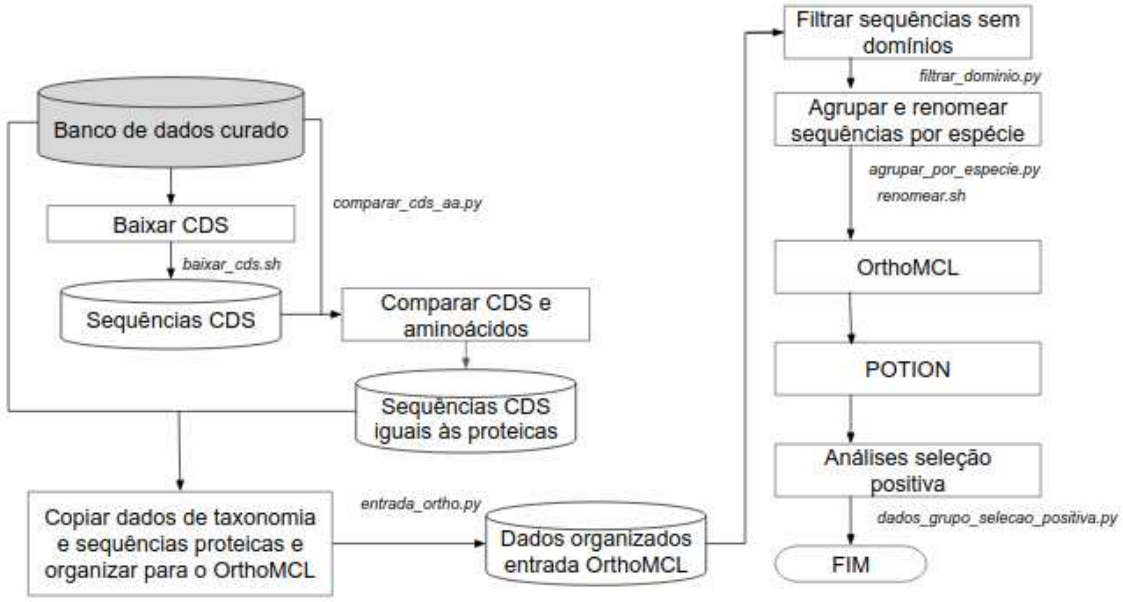

Figura 15 - Fluxograma de análise de seleção positiva.O cilindro em cinza representa o banco de dados filtrado obtidos como produto da seção 4.3. As caixas em branco representam os processos de obtenção das CDS e análise de seleção positiva. Os textos ao lado das caixas foram os scripts utilizados para realizar o processo.

4.7. Modelagem das sequências com evidência de seleção positiva

Para a predição da estrutura 3D, foi utilizado o servidor *online* do Swiss - model (WATERHOUSE *et al.*, 2018). A predição realizada foi a baseada em homologia, onde as sequências foram utilizadas para procurar modelos que pudessem ser utilizadas na homologia. Foram selecionados os melhores modelos com maior cobertura de alinhamento e similaridade de sequência. Para os modelos que obtiveram boa cobertura, analisou-se a resolução da estrutura, buscando as de melhor resolução. Em seguida, as estruturas foram preditas baseadas nesses modelos escolhidos. Para visualização e construção da imagem do melhor modelo, utilizou-se o programa VMD version 1.9.4a38 (HUMPHREY *et al.*, 1996). Com o modelo em mãos foi possível avaliar a localização espacial do sítio selecionado positivamente, conseguindo-se verificar a proximidade da superfície da estrutura e comparar essas informações com dados da literatura.

5. RESULTADOS

5.1. Banco de dados para trabalho

O banco de dados é formado pelas sequências de aminoácidos que englobam as enzimas descritas no CAZy nas categorias AA3 até a AA16. O banco é formado também pelas tabelas que contêm as informações encontradas no CAZy e os dados taxonômicos referente a cada sequência baixada. Os dados foram averiguados manualmente para confirmar a integridade das informações obtidas, para que fosse dado continuidade aos trabalhos. As tabelas e sequências foram organizadas para cada família, no caso da AA3 e AA5 para cada subfamília, individualmente. A maioria das sequências apresenta o código de depósito do banco GenPept (NCBI).

As sequências baixadas foram organizadas em arquivos no formato Fasta. Vale lembrar que algumas enzimas que constavam no banco de dados CAZy não apresentavam nenhum código de depósito, o que impossibilita a obtenção da sequência. Essas enzimas sem o código de depósito foram desconsideradas na construção do banco de sequências. Um resumo dos números de sequência em cada etapa de processamento do banco encontra-se na Tabela 1.

A análise das sequências repetidas mostram que diversas famílias apresentam sequências idênticas, mas com identificadores diferenciados. Averiguando algumas sequências manualmente, notou-se que esse fenômeno pode ser explicado devido a existência de trabalhos de sequenciamento do mesmo organismo ou de organismos próximos, o que resulta na identificação de sequências iguais. O exemplo mais dramático ocorreu com a família AA6, na qual permaneceram cerca de 45% das sequências baixadas, enquanto que as demais famílias permaneceram acima de 86%. Dois eventos de repetição em organismos diferentes também foram encontrados.

O número de sequências fúngicas dentro das famílias é outro ponto a se notar, como na família AA7, que apresentou cerca de 31% apenas de sequências fúngicas após a retirada das sequências com múltiplas cópias. A família AA15 não apresentou sequências fúngicas, porém esses dados já eram previstos, visto que a primeira integrante da AA15, descrita no trabalho de Sabbadin (SABBADIN *et al.*, 2018), tinha sido identificado em insetos e as análises de Blast realizadas no mesmo trabalho não tinha encontrado sequências pertencentes a família AA15 em espécies fúngicas. As famílias AA4, AA11 até a AA16, com exceção da AA15, foram as únicas famílias que apresentaram apenas sequências fúngicas e demais

eucariotos, sendo que as famílias AA4 e AA11 são descritas no CAZy sendo encontradas em bactérias.

Tabela 1 - Número total de sequências em cada etapa.

Família	N° de proteínas CAZy	N° sequências de proteínas baixadas	N° sequências de proteínas baixadas sem repetição	N° sequências de proteínas baixadas sem repetição de fungos
AA3	1033	1026	973	600
AA4	27	26	25	25
AA5	192	192	186	130
AA6	499	498	225	170
AA7	166	165	162	50
AA8	102	102	99	99
AA9	451	451	427	422
AA10	9	9	9	5
AA11	103	100	99	99
AA12	45	43	37	37
AA13	19	19	19	19
AA14	16	16	16	16
AA15	213	211	210	0
AA16	29	29	25	25

5.2. Sequências repetidas

Foram encontradas repetições internas em algumas famílias, sendo que apenas nas famílias AA10, AA13 e AA14 não apresentaram repetições. Vale ressaltar que algumas sequências repetiram-se muitas vezes, tendo mais de duas sequências iguais com códigos de depósito diferentes. O *script* autoral *remove_seq_duplicada.py* (Apêndice B) deixava apenas um único exemplar de cada enzima, porém, duas exceções foram encontradas, pois eram enzimas repetidas pertencentes a organismos diferentes. As exceções foram compostas pela sequência ANZ77782.1 idêntica à AAQ99151.1 ambas do grupo AA3, com os dois organismos tendo a classificação em comum mais elevada ao nível da ordem: Saccharomycetales e a sequência ABT35335.1 era idêntica à CEF78472.1, ambas do grupo AA9, em que as duas sequências tinham a classificação em comum ao nível de gênero: *Fusarium*. Essas 4 sequências pertenciam a organismos distintos, dessa forma essas 4 sequências foram mantidas dentro dos grupos. Em questão da repetição entre os grupos, vale ressaltar que já foi descrito a família AA8, contendo sequências das famílias AA3 e AA12 devido a presença do domínio citocromo, exceto isso, nenhuma outra repetição foi encontrada (LEVASSEUR *et al.*, 2013).

5.3. Irregularidades na obtenção dos dados de táxons

Foram encontradas irregularidades na obtenção dos dados taxonômicos que apareciam como última classificação nas subcategorias AA3, AA6 e AA9. Na subcategoria AA3 foram encontradas três espécies de *Cândida* que apresentavam a taxonomia como última classificação "*Ogataea*-*Candidaclade*" e duas espécies de *Penicillium* que recebiam como última classificação "*Penicilliumchrysogenumspeciescomplex*". Na subcategoria AA6 havia duas espécies de *Cândida* que recebiam a classificação "*Clavispora*-*Candidaclade*" e na subcategoria AA9 uma espécie de *Heterobasidion* que recebia a classificação de "*Heterobasidionannosumspeciescomplex*". Foram encontradas irregularidades na obtenção dos dados taxonômicos no meio de dois níveis taxonômicos em diversas famílias, sendo que todas essas incongruências eram resultantes de *incertaesedis*, presentes em diferentes níveis de taxonômicos. As taxonomias corretas foram adicionadas após a retirada das sequências não-fúngicas.

5.4. Avaliação dos gráficos

Foi gerado os gráficos de isoformas, tamanho de sequência e distribuição taxonômica para cada família individualmente. Por questão de estética e praticidade do trabalho, os

gráficos foram acrescentados nos apêndices (Apêndice H) e uma descrição com resultados pertinentes foram acrescentadas nesta seção.

A subfamília AA3sub1 apresentava 77,8% das sequências Ascomycota, 18,2% a Basidiomycota e 4% pertenciam a fungos não-cultiváveis (Figura 20). Dentro dessa subfamília, 96% das sequências tinham classificação até ao nível de gênero com 36 gêneros distintos (Figura 21) e 58,59% das sequências ao nível de espécie, com 20 espécies diferentes (Figura 22). O gênero mais frequente foi *Fusarium* com 15 sequências. A espécie de fungo com maior número de isoformas era o *Botrytiscinerea*, com 7 isoformas distintas (Figura 19). Quanto ao tamanho das sequências, foi encontrado dois picos principais próximo ao tamanho de 550 e 800 aminoácidos (Figura 23), justificável devido à presença ou não do domínio citocromo e domínio de interação ao substrato. Também pode ser visto picos com valores baixos resultantes da existência de fragmentos dentro do banco.

A subfamília AA3sub2 apresentou 23,2% de sequências Basidiomycota e 76,8% Ascomycota (Figura 25). Todas as sequências apresentaram classificação ao nível de gênero com 43 gêneros diferentes (Figura 26) e 66,76% sequências apresentaram classificação ao nível de espécie, com um total de 27 espécies (Figura 27). O gênero mais frequente é o *Aspergillus* com 69 sequências. *Zymoseptoriatritici* apresentou o maior número de isoformas, tendo 38 isoformas diferentes (Figura 24). *Aspergillusniger*e *Botrytiscinerea* também apresentaram um elevado número de sequências com 26 e 24 representantes, respectivamente. A maioria das sequências apresentam em torno de 500 a 700 aminoácidos, com uma faixa de variação restrita, porém pode-se notar picos nas região de 200 aminoácidos, devido a existência de fragmentos de sequência dentro do banco (Figura 28).

A família AA3sub3 apresentou 7,7% das sequências Basidiomycota e 92,3% Ascomycota (Figura 30). Todas as sequências apresentavam classificação ao nível de gênero com 23 gêneros diferentes (Figura 31). Das 65 sequências apenas 55,38% apresentaram classificação no nível de espécie, formada por 17 espécies distintas (Figura 32). O gênero *Ogataea* foi o mais presente, com 15 sequências distintas. O clado *Ogataea*-*Candida*e a espécie *Zymoseptoriatritici*, apresentaram 5 isoformas, sendo as espécies com mais isoformas no banco (Figura 29). A maioria das sequências variou entre 600 e 700 aminoácidos, tendo um pico em 650. O gráfico também apresentou um pico ao redor de 250 aminoácidos decorrente da existência de fragmentos no banco (Figura 33).

A família AA3sub4 apresentou 72,2% de sequências Basidiomycota e 27,8% Ascomycota (Figura 34). Todas as sequências apresentaram classificação ao nível de gênero

com 11 gêneros diferentes (Figura 35) e apenas 27,78% sequências apresentaram classificação ao nível de espécie com 5 espécies diferentes (Figura 36). O gênero *Trametes* foi o mais frequente com 5 sequências. Não foram encontradas isoformas nessa família de enzimas. As sequências da AA3sub4 apresentaram tamanhos de sequência em torno de 620 aminoácidos com a maior sequência tendo 653 aminoácidos (Figura 37).

A família AA4 apresentou em sua maioria Ascomycota apresentando 88% sequências contra apenas 12% Basidiomycota (Figura 39). Todas as sequências apresentaram classificação ao nível de gênero, presentes em 10 gêneros diferentes (Figura 40) e 68% sequências apresentaram ao nível de espécie com 12 espécies diferentes (Figura 41). O gênero mais presente era o *Podospora* com 8 sequências, seguido de *Fusarium* com 4. A espécie *Podospora anserina* apresentou 4 isoformas (Figura 38). O tamanho da sequência variou entre 525 e 625, tendo seu maior pico em torno de 600 O gráfico de tamanho também apresentou um pico decorrente da existência de fragmentos no banco, em torno de 350 aminoácidos (Figura 42).

A subfamília AA5sub1 apresentou 77% de sequências Basidiomycota, 21,6% Ascomycota e 1,4% Mucoromycota (Figura 44). Todas as sequências apresentaram classificação ao nível de gênero e 27 gêneros diferentes estão presentes (Figura 45). Apenas 58,11% sequências apresentaram classificação ao nível de espécie, com 22 espécies diferentes (Figura 46). Os gêneros *Sporisorium* e *Serendipita* foram os mais encontrados, com 9 e 8 sequências respectivamente. Em questão das isoformas, *Serendipita indica* apresentou 8 isoformas enquanto que *Sporisoriumreilianum* apresentou 6 isoformas (Figura 43). Em questão do tamanho das sequências, foi encontrado um pico em torno de 700 aminoácidos, porém sequências com mais de 1000 aminoácidos também ocorrem (Figura 47). Isso pode ser explicado pela repetição de domínios WSC que apresentava variação no número de cópias fazendo com que os picos se distribuíssem no gráfico. Alguns fragmentos também aparecem no gráfico e são descritos adiante no trabalho.

A família AA5sub2 apresentou todas as sequências Ascomycota e todas tem classificação ao nível de gênero e 86,96% das sequências ao nível de espécie, apresentando 13 gêneros (Figura 49) e 16 espécies diferentes (Figura 50). O gênero *Fusarium* foi o mais frequente, com 20 sequências diferentes. Em questão de isoformas, as espécies *Fusariumfujikuroi*, *Fusariumgraminearum*, *Botrytiscinerea*e*Zymoseptoriatritici* apresentaram 4 isoformas (Figura 48). Em questão de tamanho, a sequência apresenta,em média, entre 600 e 700 aminoácidos, mas chegou a apresentar sequências maiores (Figura 51). Os picos com

maior número de aminoácidos podem ser explicados devido à existência de alguns domínios extras encontrados durante a análise de domínios, mas como a família não apresentou um domínio característico caracterizado pelo Pfam não é possível demonstrar a existência de domínios duplicados a partir dos dados analisados.

A família AA6 teve 14,1% de sequências Basidiomycota, 82,4% Ascomycota e 3,5% Mucoromycota (Figura 53). Todas as sequências apresentaram classificação ao nível de gênero contendo 48 gêneros distintos (Figura 54) e 68,82% sequências ao nível de espécie com 40 espécies diferentes (Figura 55). O gênero mais predominante foi o *Saccharomyces* com 30 sequências, seguido de *Candida* com 15 e *Zygosaccharomyces* com 11 sequências. A espécie com maior número de isoformas foi a *Saccharomyces cerevisiae* com 27 sequências diferentes (Figura 52). A maioria das sequências apresentou entre 200 e 250 aminoácidos, sendo poucas sequências maiores que esse valor (Figura 56). A família apresentava algumas sequências que apresentavam indícios de ser fragmentos, porém essas não conseguiam ser facilmente identificadas no gráfico. Algumas sequências apresentaram domínios duplicados, podendo justificar as sequências com cerca de 350 aminoácidos.

A família AA7 apresentou 88% das sequências Ascomycota e apenas 12% Basidiomycota (Figura 58). Todas apresentaram classificação ao nível de gênero com 21 gêneros diferentes, sendo *Fusarium* e *Podospora* os gêneros mais frequentes com 10 sequências cada (Figura 59). Foram encontradas 62% sequências classificadas ao nível de espécie com 15 espécies distintas (Figura 60). O maior número de isoformas foi encontrado em *Fusariumgraminearum*, *Fusariumfujikuroi*, *Pyriculariaoryzae,Serendipita indica*queapresentaram 3 isoformas diferentes cada (Figura 57). A maioria das sequências apresentaram 500 aminoácidos, mas foi encontrado sequências com mais de 700 aminoácidos, podendo ser explicado pela identificação de domínios de interação à carboidrato em algumas sequências (Figura 61). Foi possível identificar picos referentes à existência de fragmentos no banco. O gráfico de isoformas se encontra na Figura 57.

A família AA8 apresentou 80,8% das sequências Ascomycota e apenas 19,2% Basidiomycota (Figura 63). Todas as sequências apresentaram classificação ao nível de gênero, apresentando 33 gêneros diferentes (Figura 64), porém apenas 59,6% sequências apresentaram classificação ao nível de espécie com 18 espécies diferentes (Figura 65). O gênero mais frequente era o *Podospora* com 15 sequência, seguido do *Fusarium* com 11. O maior número de isoformas se encontra no *Podospora anserina* e *Pyriculariaoryzae* com 8 e 7 sequências respectivamente (Figura 62). Dois picos foram encontrados na tabela de

tamanho, sendo um em cerca de 200 aminoácidos e outro em torno dos 800 (Figura 66). Isso se justifica pela família AA8 conter o domínio citocromo livre e também sequências de CDH da família AA3sub1 e membros da família AA12 que apresentam múltiplos domínios.

A família AA9 apresentou 13% das sequências como Basidiomycota, 82,5% Ascomycota e 4,5% Chytridiomycota (Figura 68). Todas as sequências apresentaram classificação ao nível de gênero, com 49 gêneros diferentes (Figura 69). Em diversidade de espécies, foram encontradas apenas 25 espécies distintas, com 71,33% sequências que tiveram esse nível de classificação (Figura 70). O gênero mais frequente foi *Podospora* com 59 sequências, seguido de *Fusarium* com 41. O maior número de isoformas foi de *Podospora anserina* com 33 sequências seguido de *Arthrobotrysoligospora* com 26 (Figura 67). A maioria das sequências apresentaram em torno de 250 aminoácidos, poucas sequências apresentaram um tamanho maior, como será visto mais à frente, devido provavelmente a domínios duplicados (Figura 71).

A família AA10 apresentou apenas 5 sequências, estando presente apenas em Basidiomycota e com todas as sequências apresentando classificação em nível de gênero em 3 gêneros diferentes (Figura 72). Apenas 60% das sequências apresentaram classificação ao nível de espécie em duas espécies diferentes (Figura 73). Os gêneros encontrados foram *Melanopsichium* e *Ustilago* com uma sequência cada e *Sporisorium* com três. As espécies encontradas foram *Sporisoriumreilianum* e *Melanopsichiumpennsylvanicum*. Apenas a espécie *Sporisoriumreilianum*apresentou isoformas com duas isoformas diferentes. As sequências apresentaram tamanho em torno de 296 e 332 aminoácidos, tendo tamanhos bem próximos (Figura 74).

A família AA11 apresentou todas com classificação ao nível de gênero com 18 gêneros diferentes (Figura 77). Os filos encontrados foram Ascomycota com 93,9% sequências, Basidiomycota com 4%, Zoopagomycota e Mucoromycota com 1% cada (Figura 76). Dentreas sequências, 85,86% apresentaram classificação em nível de espécie com 17 espécies distintas (Figura 78). O gênero mais presente foi *Zymoseptoria* com 19 sequências. O organismo com maior número de isoformas foi o *Zymoseptoriatritici* que apresentou 19 isoformas diferentes (Figura 75). O tamanho das sequências teve seu maior pico em 400 aminoácidos com uma distribuição ao redor desse valor. Algumas sequências contendo entre 870 a 1200, porém, foram poucas (Figura 79). Como não foi encontrado domínios característicos para a família no Pfam, não podemos identificar a existência de domínios duplicados a partir do que foi analisado.

A família AA12 teve todas as sequências apresentando classificação em nível de gênero, distribuído em 16 gêneros diferentes (Figura 82). O filo mais presente foi o Ascomycota com 91,9% sequências e o Basidiomycota com 8,1% (Figura 81). As sequências apresentaram 16 espécies diferentes, sendo que 81,08% sequências apresentaram classificação ao nível de espécie (Figura 83). O gênero mais frequente foi o *Podospora* com 8 sequências e *Fusarium* com 5. As espécies *Podospora anserina* e *Pyriculariaoryzae* apresentaram 4 isoformas cada (Figura 80). O tamanho das sequências se apresentava entre 400 e 800 aminoácidos, tendo seu maior número de sequências entre 400 e 500 e poucos indivíduos acima desse valor (Figura 84). Um domínio característico da família não foi encontrado, porém, algumas sequências apresentaram domínios citocromo e domínios de interação à carboidratos, que podem justificar a diferença de tamanho.

A família AA13 teve todas as sequências pertencentes às Ascomycota. Todas as sequências apresentaram classificação em nível de gênero com 11 gêneros distintos (Figura 86) e 94,74% apresentaram classificação de espécie com 15 espécies diferentes (Figura 87). O gênero mais presente foi o *Aspergillus* com 4 sequências, em seguida, *Fusarium* com 3 sequências. As espécies *Aspergillusnidulans*, *Botrytiscinerea*, *Penicilliumrubens*todos apresentaram 2 isoformas (Figura 85). Os tamanhos da sequência tiveram o maior pico em 250 aminoácidos, mas muitas sequências ficaram entre 325 e 425 aminoácidos (Figura 88). A existência de domínios de interação a carboidratos pode justificar a disparidade de tamanho.

A família AA14 apresentou 75% das sequências Basidiomycota e 25% de Ascomycota (Figura 89). Todas as sequências apresentaram classificação em nível de gênero, com 6 gêneros distintos (Figura 90). Apenas 37,5% sequências apresentaram classificação em nível de espécie, estando distribuídas em 5 espécies diferentes (Figura 91). Foi observado que a única espécie a apresentar isoformas foi a *Cryptococcusneoformans* com duas sequências apenas. A maioria das sequências apresentaram tamanho entre 300 e 450 aminoácidos, porém sequências em torno de 545 e 860 também foram encontradas (Figura 92). A família AA14 não apresentou domínio característico para a enzima, dificultando a verificação de fragmentos ou domínios duplicados, mas apresentou sequências WSC que poderima ter duplicações e justificar a disparidade de tamanho.

A família AA16 apresentou 12% de sequências Basidiomycota e 88% de Ascomycota (Figura 94). Todas as sequências apresentaram classificação em nível de gênero distribuídas em 12 gêneros diferentes (Figura 95) e 84% sequências ao nível de espécie, em 16 espécies distintas (Figura 96). As espécies *Aspergillusoryzae* apresentou duas isoformas e

Pyriculariaoryzae e *Thermothelomycesthermophilus* apresentaram três cada (Figura 93). O tamanho das sequências variaram de 200 a 400 aminoácidos,com uma concentração em torno de 300 aminoácidos (Figura 97). A família apresentou sequências fragmentadas, mas não eram facilmente verificadas no gráfico.

Analisando os gráficos, pode-se ver que em 13 das 17 enzimas trabalhadas, o número de sequências pertencentes ao Filo Ascomycota era maior do que as pertencentes ao Filo Basidiomycota. O acesso dos IDs e a forma como são armazenadas no NCBI nos permite verificar algumas situações, utilizando para isso, os esquemas de como as sequências são nomeadas (OBS: informações obtidas de https://www.ncbi.nlm.nih.gov/Sequin/acc.html e https://www.ncbi.nlm.nih.gov/refseq/about/). Nos dados foram encontrados IDs nomeadas como início B, C, S, V, A, Q, G, E, K e XP. Sabe-se que IDs começadas com G, E, K e XP são WGS (Wholegenomesequencing) Protein ID. Os IDs começados com as demais siglas não são necessariamente WGS (Wholegenomesequencing) Protein ID, podendo ser oriundas de outros projetos de sequenciamento diversos como de regiões específicas do DNA, como no caso da sequência AAA34321.1, ou mRNA, como no caso da sequência AAA32695.1, assim como também de projetos genômicos como no caso da sequência BAP70113.1. Atenção especial pode ser dada para as sequências iniciadas com a sigla XP, pois são oriundos de anotações automáticas do NCBI de genomas da base RefSeq.

Fazendo-se uma contagem das sequências, pode-se ver que aproximadamente 12,26% das sequências obtidas após as etapas de filtragem eram de início G, E, K e XP, sendo que dessas sequências 93,75% eram de Ascomycota e 6,25% eram de Basidiomycota e todas as sequências de início XP eram de Ascomycota. Dentre o total do banco de sequências, 79,32% eram Ascomycota e 18,74% eram de Basidiomycota, os 1,94% das sequências eram de filos distintos ou sem classificação taxonômica. Esses números podem mostrar que existe um maior número de sequências Ascomycota devido, como um dos motivos, ao maior número de estudos em Ascomycota ao invés dos Basidiomycota, apesar de os fungos Basidiomycota serem considerados os melhores degradadores de lignina (BUGG *et al.*, 2011).

5.5. Resultado da análise dos domínios

O resultado dos domínios localizados pelo Pfam estão descritos na Tabela 2. Os domínios encontrados poderiam ser descritos como domínios comuns entre as sequências que compunham a família, caso estivessem presentes em várias sequências, sendo característico da enzima trabalhada e estando de acordo com a literatura (Apêndice G).

Também foi descrito casos em que o domínios identificados eram condizentes com a literatura, mas não foram amplamente encontrados em todas as sequências do grupo. Um caso que descreve isso é a da subfamília AA5sub2, sendo descrita composta por múltiplos domínios Kelch, porém muitas sequências apresentaram diferentes domínios Kelch, além de outras não conter nenhum domínio Kelch identificado.

Uma última situação descrita são domínios que foram identificados pelo Pfam e quando verificados na literatura não tinham relação com a enzima que compõe a família. Um exemplo é a família AA9, que apresentou o domínio Lactamase_B_2 encontrado em enzimas degradadoras de antibiótico.

Os domínios considerados comuns e característicos das enzimas que compõem as famílias estavam de acordo com trabalhos estruturais de proteínas e revisões da literatura (Apêndice G). Para cada domínio identificado foi feito um estudo e uma descrição com o máximo de informações encontradas contidos no Apêndice F e a revisão mostrando os domínios encontrados em trabalhos estruturais se encontra no Apêndice G.

Tabela 2 - Domínios encontrados para cada família.

Família	DC	DN-C	DI
AA3sub1	CDH-cyt GMC_oxred_N GMC_oxred_C CBM_1	-	-
AA3sub2	GMC_oxred_N GMC_oxred_C	CBM_1	DSBA Pkinase RWD HET PhyH Asp
AA3sub3	GMC_oxred_N GMC_oxred_C	-	-
AA3sub4	GMC_oxred_N GMC_oxred_C	-	FAD_binding_3
AA4	FAD_binding_4 FAD-oxidase_C	-	-
AA5sub1	Glyoxal_oxid_N DUF1929	WSC Chitin_bind_1	Podoplanin

AA5sub2	DUF1929	WSC Glyoxal_oxid_N Kelch_1 Kelch_4 Kelch_6	F5_F8_type_C PAN_1
AA6	FMN_red	-	Flavodoxin_1
AA7	FAD_binding_4 BBE	CBM_1 Chitin_bind_1	Amidohydro_2
AA8	CDH-cyt	GMC_oxred_N GMC_oxred_C CBM_1	-
AA9	Glyco_hydro_61 CBM_1	Chitin_bind_1	CFEM Lactamase_B_2
AA10	LPMO_10	-	-
AA11	-	Glyco_hydro_61 LPMO_10	-
AA12	-	CBM_1 CDH-cyt GSDH	SGL
AA13	LPMO_10	CBM_20	-
AA14	-	WSC	-
AA16	LPMO_10	-	-

A tabela 2 apresenta os domínios encontrados pelo Pfam com valor estatístico menor ou igual a $9.9e^{-5}$. Onde DC - Domínios característicos, corresponde a domínios comuns a uma grande quantidade de sequências da família analisada, estando associados pela literatura à funcionalidade da enzima; DN-C - Domínios não-característicos, correspondem a domínios encontrados em poucas sequências da família analisada, mas estão associados pela literatura a funcionalidade da enzima; DI - Domínios incomuns, corresponde a domínios pouco comuns nas sequências e que não são descritos na literatura envolvidos na atividade da família analisada.

Ocorreram casos em que domínios considerados característicos de uma família estavam ausentes ou apresentavam baixo valor estatístico. Em muitos casos em que um domínio estava ausente a sequência tinha na sua descrição no CAZy um indicativo que se tratava de um fragmentos.

Com a análise dos domínios foi possível identificar enzimas com domínios duplicados, nos quais esses domínios tinham características de tamanho e alinhamento que mostravam que realmente foram duplicados. Em outros casos foram encontrados domínios identificados de forma fragmentada, sendo que esses domínios, quando avaliados com os demais domínios de outras sequências da família que apresentaram apenas um domínio identificado, alinhavam-se na mesma região e dentro do mesmo intervalo. Dessa forma, esses domínios identificados de forma fragmentada poderiam corresponder a domínios únicos que foram mal identificados, mas que podem corresponder a domínios únicos devido à análise do alinhamento das sequências (Tabela 3).

Tabela 3 - Famílias que tiveram domínios faltantes ou que foram localizados duplicados.

Família	DF	SDD	SDF
AA3sub1	Sim	Não	Sim
AA3sub2	Sim	Sim	Sim
AA3sub3	Sim	Não	Sim
AA3sub4	Sim	Não	Não
AA4	Sim	Não	Não
AA5sub1	Sim	Sim	Sim
AA5sub2	Não	Sim	Não
AA6	Sim	Sim	Não
AA7	Sim	Não	Não
AA8	Não	Não	Não
AA9	Sim	Sim	Sim
AA10	Não	Não	Não
AA11	Não	Não	Não
AA12	Não	Não	Não
AA13	Sim	Não	Não
AA14	Não	Sim	Não
AA16	Sim	Não	Não

A tabela 3 apresenta quais famílias tiveram sequências com domínios faltantes ou duplicados. Onde DF - Domínios faltantes: apresentou alguma sequência que não teve um domínio característico identificado ou apresentou esse domínio com baixo valor estatístico; SDD - sequências com domínios duplicados, apresentou alguma sequência com domínios duplicados e que tinha características de alinhamento e de tamanho que mostravam que o domínio era duplicado; SDF - Sequência com domínios fragmentados, os quais apresentaram alguma sequência que teve vários domínios identificados, mas seu alinhamento correspondia a um domínio completo nas demais sequências.

5.5.1. Resultados dos domínios na família AA3sub1

Algumas sequências não apresentaram domínio CDH-cyt, mas nãohá a necessidade de ocorrer esse domínio para a enzima ter atividade. Todas as sequênciasapresentaram domínio GMC_oxred_N, o qual é um domínio de ancoragem de flavina e apresenta atividade catalítica.

O domínio GMC_oxred_C foi encontrado e é descrito pelo Pfam como região de ligação a esteroides. Buscando mais informações sobre esse domínio foi encontrado o trabalho (ZÁMOCKÝ *et al.*, 2004) que mostra uma proximidade evolutiva entre as colesterol oxidase e as CDH. Analisando os dados da estrutura da enzima CDH no PDB "1NAA", viu-se que os aminoácidos que interagem com o substrato estão nessa região, chamada de GMC_oxred_C, sendo provavelmente referente a mesma região de ligação do substrato.

Algumas sequências não apresentaram o domínio GMC_oxred_C. No site do Uniprot foi descrito que o Pfam considera os domínios CDH-cyt, GMC_oxred_N e GMC_oxred_C pertencentes a proteína CDH (BATEMAN, 2019; UNIPROT, 2019a).

As sequências CCD52080.1 e AEO61303.1 apresentaram tamanho de sequência dentro da faixa de 500 à 600 aminoácidos, porém a sequência CCD52080.1 apresentou apenas os domínios GMC_oxred_N e CDH-cyt, enquanto que a sequência AEO61303.1 apresentou o valor de E-value muito maior que o valor de corte para o domínio GMC_oxred_C. As demais sequências que não apresentaram o domínio GMC_oxred_C apresentavam tamanho menores de 200 aminoácidos, com exceção da sequência ADV29795.1 com tamanhona faixa de 500 a 600 aminoácidos, mas foi descrita como *fragment* pelo banco de dados Cazy. A sequência CBI59551.1 apresentou domínios identificados como duplicados, mas que no alinhamento correspondia a um único domínio.

O domínio de ligação CBM_1 pode estar presente ou não, visto que outros possíveis mecanismos de interação com a celulose podem existir, utilizando o domínio CBM ou por

aminoácidos específicos na superfície da enzima (HENRIKSSON *et al.*, 2000; ZÁMOCKÝ *et al.*, 2004; SÜTZL *et al.*, 2019). Nenhum domínio duplicado foi encontrado. O tamanho de todos os domínios foram semelhantes e a posição no alinhamento também foram parecidas, indicando que a região do domínio é preservada.

5.5.2. Resultados dos domínios na família AA3sub2

Todas as sequências maiores que 400 aminoácidos apresentaram domínios funcionais GMC_oxred_N e GMC_oxred_C. Algumas sequências menores que 400 aminoácidos foram descritos com o termo *fragment* no banco Cazy e não apresentaram algum dos dois domínios GMC. Algumas sequências apresentaram domínios extras, que não são normais de serem encontrados nas enzimas pertencentes a essa família. Esses domínios se apresentaram restritos a determinadas sequências. O domínios GMC_oxred_N apresentou alinhamento com domínios próximos, porém algumas sequências tiveram domínios maiores ou menores em relação a maioria observada. O domínio GMC_oxred_C ficou em uma faixa restrita de tamanho no alinhamento das sequências.

Algumas sequências apresentaram um mesmo domínio identificado duas vezes, apresentando tamanho de domínio identificado menordoque as demais sequências. Esses domínios duplos correspondiam ao domínio único das outras enzimas quando avaliados no alinhamento. Esse fenômeno pode ter sido provocado por um domínio com região com similaridade baixa, fazendo com que o domínio seja identificado em duas regiões. Esse fenômeno ocorreu apenas para o domínio GMC_oxred_N nas sequências EAA61600.1, CDP24859.1, CDP27647.1, CDP31727.1, VBB74103.1, EAQ70756.1, CCA67046.1, SMR56796.1, SMQ51093.1 e SMQ53215.1.

A sequência EGX43942.1 apresentou além dos domínios GMC, o domínio de ligação a carboidratos CBM_1. Como pode ser visto na CDH, um módulo de ligação a celulose parece ser interessante para a atividade da enzima, mesmo não sendo obrigatório para a sua atividade. Experimentos mostraram que a adição de módulos CBM em Glucooligosaccharide Oxidase, pertencente à família AA7, fez com que a atividade da enzima aumentasse em celulose amorfa regenerada e outros carboidratos (FOUMANI *et al.*, 2015). Talvez esse módulo possa melhorar a atividade da enzima junto ao substrato.

A sequência EAA61740.1 apresentou dois domínios peculiares dentro de suas sequências, comoos domínios DSBA e Pkinase. Ela apresentou os domínios GMC esperados e dentro da faixa do alinhamento das demais. Os tamanhos dos domínios foram semelhantes com outras sequências do banco, além do intervalo de alinhamento onde ocorriam. A ligação

desses módulos poderia ser funcional e facilitar a regulação da produção das enzimas, ou mesmo existir uma interação entre eles, necessitando de mais estudos para mostrar a existência de funcionalidade.

A sequência AYO41662.1 apresentou apenas o domínio RWD a mais do que o esperado. A literatura indica esse domínio como um domínio de interação com outras proteínas, não deixando claro as vantagens oferecidas por esse domínio para o funcionamento da proteína (ALONTAGA *et al.*, 2015). Os demais domínios GMC estão alinhados junto com o intervalo das outras sequências do banco e apresentam tamanhos semelhantes também, demonstrando características de conservação.

A sequência CDP31201.1 apresentou o domínio HET. O tamanho da sequência do domínio é de 148 aminoácidos, bem próximos ao valor normalmente conservado para o domínio (ESPAGNE *et al.*, 2002). As regiões dos domínios GMC se alinham na mesma região dos demais domínios das outras sequências e com tamanho de domínio dentro da faixa das demais enzimas. A real vantagem da inserção do HET na sequência não é clara, mas no trabalho de Wichmann*et al.*, (2008), foi demonstrado a existência de uma sequência de *Pseudomonas syringae* semelhante ao HET e, quando expresso em*Neurospora crassa*,gerava fenótipo semelhante à incompatibilidade heterocariótica. Eles também verificaram uma certa vantagem de crescimento da *P. syringae*quando associado ao fungo.

A sequência CED85556.1 apresentou o domínio PhyH, que atua na hidroxilação de moléculas orgânicas. A sequência apresentou domínios GMC que tinham características normais de tamanho e alinhamento com as demais sequências do banco. Não foi encontrada nenhuma informação que relacione o domínio PhyH com a degradação de lignina. Os dados indicam um provável ganho de função da enzima no citoplasma.

O domínio Asp foi encontrado em 3 sequências, SMR51427.1, SMY23191.1 e SMQ49497.1, em que todos apresentaram os domínios GMC com características de tamanho e alinhamento normais com as demais sequências. As enzimas Asp são amplamente distribuídas entre os fungos esãopouco conhecidas suas funções fisiológicas (REVUELTA *et al.*, 2014). Enzimas com o domínio Asp podem ser encontrados no secretoma do fungo *Phanerochaetechrysosporium* durante condições de baixa concentração de nitrogênio e carbono (WYMELENBERG *et al.*, 2006), mostrando que esse domínio da enzima pode ser funcional durante o processo de degradação de lignina em condições bióticas de estresse. Não foram encontrados domínios duplicados.

5.5.3. Resultados dos domínios na família AA3sub3

Asubfamília AA3sub3 consiste na enzima álcool oxidase. Essa enzima é composta por dois domínios, um sendo o de ligação ao FAD e o outro o de interação ao substrato (KOCH *et al.*, 2016). Todas as sequências apresentaram GMC_oxred_N e apenas as sequências BAF63432.1, BAF63445.1, BAF63440.1 não apresentaram o domínio GMC_oxred_C. As sequências que não apresentaram domínio GMC_oxred_C tinham tamanhos menores em relação às sequências com todos os domínios e eram descritas como *fragment* pelo CAZy. As sequências apresentaram tamanhos de domínio e intervalo de alinhamento próximos, tanto para o domínio GMC_oxred_N, quanto para GMC_oxred_C. Algumas sequências apresentaram um mesmo domínio identificado duas vezes, apresentando tamanho de domínio identificado menor que as demais sequências. Esses domínios duplos correspondiam ao domínio único das outras enzimas quando avaliados no alinhamento. As sequências SMR47069.1 e SMQ47288.1 apresentaram essa característica para o domínio GMC_oxred_N e a sequência CBF89219.1 para o domínio GMC_oxred_C.

5.5.4. Resultados dos domínios na família AA3sub4

Todas as sequências apresentaram domínio GMC_oxred_C, mas quando se trata do domínio GMC_oxred_N, apenas as sequências CBF82183.1, ACM47528.1, ALJ82907.1, AVP27637.1 apresentaram essedomínio com valores E-value menor ou igual a $9.9e^{-5}$. As demais sequências apresentaram a presença do domínio GMC_oxred_N, mas todas ficaram com E-value mais positivos. Os tamanhos dos domínios GMC_oxred_N e GMC_oxred_C encontrados, também foram menores em relação às outras subfamílias, mostrando que o Pfam não foi capaz de caracterizar todo o domínio da enzima Pyranose oxidase. Isso pode ser explicado pela baixa relação com as demais enzimas da família GMC, além da baixa quantidade de enzimas encontradas em banco de dados, como sugerido pelo trabalho de Sützl*et al.* (2019), que utilizou sequência de enzimas caracterizadas para tentar clusterizar novas sequências (SÜTZL *et al.*, 2019). Essa subfamília compreende um grupo de enzimas mais distante filogeneticamente em relação aos outros membros relatados da família GMC (SÜTZL *et al.*, 2018).

Um pedaço da sequência CCT76068.1 apresentou semelhança com um domínio FAD_binding_3, com E-value abaixo do valor de corte, porém com e-value maiores em outras 6 sequências. A região de alinhamento também foi pequena, apresentando 34 aminoácidos na sequência CCT76068.1. Esse domínio está relacionado a um domínio de ligação ao FAD, descrito no Pfam pelo código PF01494.19, podendo compreender uma

estrutura diferente de interação ao FAD. Nenhuma das sequências apresentou domínio duplicado.

5.5.5. Resultados dos domínios na família AA4

Foram encontrados apenas os domínios FAD_binding_4 e FAD-oxidase_C. A sequência SAM85895.1 não apresentou o domínio FAD_binding_4. A sequência tinha apenas 332 aminoácidos e era descrita como *fragment* pelo CAZy. A sequência CDP22417.1, mesmo tendo 547 aminoácidos, era descrita como *fragment* pelo CAZy e não apresentou o domínio FAD-oxidase_C. As sequências apresentaram tamanhos de domínio e intervalo de alinhamento próximos. A sequência EKV12372.1 apresentou um tamanho de domínio FAD_binding_4 caracterizado menor que as demais, mas com alinhamento na mesma região. O mesmo aconteceu para a sequência CAP97133.1 que apresentou um tamanho de domínio FAD-oxidase_C caracterizado menor que as demais, mas com alinhamento na mesma região.

5.5.6. Resultados dos domínios na família AA5sub1

Apenas a sequência ALL40757.1 não apresentou o domínio Glyoxal_oxid_N. Essa sequência era descrita como *fragment* pelo CAZy e consiste de uma sequência de 294 aminoácidos. O domínio DUF1929 estava presente em todas as sequências. Provavelmente os domínios Glyoxal_oxid_N e DUF1929 são correspondentes a região de interação com o cobre e de ação enzimática, como pode ser visto na análise da literatura. Algumas sequências apresentaram um mesmo domínio identificado duas vezes, apresentando tamanho de domínio identificado menor do que as demais sequências. Esses domínios duplos corresponderam ao domínio único das outras enzimas quando avaliados no alinhamento. Esse fenômeno ocorreu para o domínio Glyoxal_oxid_N. Os domínios Glyoxal_oxid_N e DUF1929 apresentaram tamanho similar nas sequências e mesma região de alinhamento mostrando conservação da posição do domínio.

Duas sequências, CAD89077.1 e CAD88906.1 apresentaram o domínio Chitin_bind_1. O domínio WSC foi encontrado em 17 sequências e apresentava múltiplos domínios na maioria delas, variando de duas repetições em ABD61574.1 até sete repetições em CCA74522.1. Essas repetições podem ter ocorrido, pois o Pfam não reconhece necessariamente o domínio inteiro, caso um domínio tenha uma região de fraca identificação, podendo ser identificado o início e o final de maior especificidade, porém é possível encontrar múltiplos domínios WSC em Glioxal oxidase, além de o fato de todos os domínios

identificados estarem na faixa de 80 aminoácidos muito próximo ao descrito no artigo de expansão do CAZy (LEVASSEUR *et al.*, 2013; VANDEN WYMELENBERG *et al.*, 2006).

Foi encontrada também uma sequência contendo uma região de identificação da família de proteínas podoplanina, presente em mamíferos e que regulam podócitos do rim. Esse domínio foi encontrado apenas na sequência SHO78428.1, com valor estatístico significativo.

5.5.7. Resultados dos domínios na família AA5sub2

Para a família AA5sub2, é mais difícil de se descrever os domínios utilizando o Pfam, visto que a região catalítica parece ser identificada por vários domínios separados. Foram identificadas 9 sequências com E-value menor ou igual a $9.9e^{-5}$, tendo uma região semelhante com o domínio Glyoxal_oxid_N, porém todas as outras mostraram regiões Glyoxal_oxid_N com E-value maior que o valor de corte. Diferentemente das sequências da família AA5sub1, que apresentaram esse domínio com cerca de 200 aminoácidos, as sequências na família AA5sub2, que tinham e-value confiável, continham em torno de 80 aminoácidos. Todas as sequências apresentaram pelo menos um domínio Kelch, mas nem todas as sequências apresentaram esse domínio com E-value significativo. O Pfam identificou três domínios Kelch diferentes, Kelch_1, Kelch_4 e Kelch_6, mas nem todas sequências apresentavam as três. Os domínios Kelch_1 e Kelch_4 se alinhavam em regiões próximas e com tamanhos semelhantes. O domínio Kelch_6 apresentou regiões de alinhamento distintos, podendo se alinhar junto do Kelch_4 ou em uma região anterior ao Kelch_1. Isso demonstra que provavelmente todas as sequências presentes na família AA5sub2 apresentam trechos do domínio ativo, formadas por isoformas de domínio Kelch. Todas as sequências apresentaram o domínio DUF1929, importante na coordenação do cobre no sítio ativo. O tamanho e região de alinhamento do domínio DUF1929 foram bem parecidos em todas as sequências.

Muitas sequências apresentaram o domínio F5_F8_type_C. Este domínio está associado com interação com a membrana fosfolipídica, no entanto, relatos desse domínio em fungos não foram encontrados. Uma relação direta desse domínio com a degradação de lignina ou funcionamento da enzima não foi encontrada. Esse domínio pode servir para ancorar a enzima na superfície da célula contendo fosfolipídeos. Algumas sequências apresentaram o domínio PAN_1, descrito na literatura com um domínio de interação proteína-proteína, proteína-carboidrato, podendo estar relacionado com a interação ao substrato. Algumas sequências apresentaram dois domínios PAN_1. Como não foi encontrado relato do tamanho do domínio, isso pode ser devido à baixa similaridade da sequência ou duplicação do

domínio. Dados adicionais não foram encontrados. A sequência XP_003719369 apresentou o domínio WSC, que está relacionado com interação a carboidratos. Já foi visto relato de uma álcool oxidase contendo esse tipo de domínio, mas pela análise dos dados, isso não é algo comum (YIN *et al.*, 2015).

5.5.8. Resultados dos domínios na família AA6

A família AA6 apresentou apenas oito sequências sem o domínio FMN_red ou com E-value acima do valor menor ou igual a $9.9e^{-5}$, sendo que duas dessas sequências eram descritas como *fragment* pelo CAZy. O domínio Flavodoxin_1 também foi encontrado, sendo estatisticamente confiável na sequência CCC71216.1. Outras cinco sequências apresentaram o domínio Flavodoxin_1, mas com baixo valor estatístico e, em quatro dessas sequências, havia o domínio FMN_red com valor estatístico aceitável, mas com regiões do domínio identificado menor que as demais sequências. A identificação da sequência CCC71216.1 como Flavodoxin_1 não é esperada, visto que, mesmo tendo estruturas tridimensionais próximas, a sequência da Flavodoxin_1 é pouco similar em relação às outras flavoproteínas, apresentando apenas algumas regiões preservadas (GRANDORI *et al.*, 1998; GRANDORI *et al.*, 1994).

A sequência SGZ57668.1 e SGZ56402.1 apresentaram o domínio FMN_red duplicado. A SGZ56402.1 apresentou um domínio de 125 e outro com 89 aminoácidos. A SGZ57668.1 apresentou domínios maiores, com 125 e 110 aminoácidos. Esse valor ficou próximo a outros valores encontrados para os domínios das demais sequências, sendo provavelmente domínios funcionais. As sequências apresentaram uma variação no tamanho do domínio FMN_red, apresentando mesma região de alinhamento. O domínio Flavodoxin_1 encontrado com valor estatístico aceitável apresentava o alinhamento juntamente com o domínio FMN_red das outras sequências. Quatro sequências que apresentavam o domínio Flavodoxin_1 com baixo valor estatístico apresentaram o domínio FMN_red estatisticamente aceitável, mas com região identificada reduzida.

5.5.9. Resultados dos domínios na família AA7

Todas as sequências analisadas apresentaram o domínio FAD_binding_4, e as sequências VBB81530.1 e SMQ49385.1 são descritas como *fragment* no banco de dados CAZy. Todas as sequências apresentaram o domínio BBE, entretanto, as sequências VBB79782.1 e VBB84781.1 apresentaram valores de E-value maiores que o valor menor ou igual a $9.9e^{-5}$.

Atenção especial deve ser dada para a sequência SMQ49385.1 que apresentou os domínios BBE e Amidohydro_2. O domínio BBE ocorre antes do domínio Amidohydro_2, contrariamente às sequências contendo FAD_binding_4 que apresentam o domínio BBE após o domínio FAD_binding_4. A SMQ49385.1 apresenta 417 aminoácidos, sendo descrita como *fragment*, mesmo assim teria tamanho de sequência que compreenderia grande parte de uma enzima funcional. O domínio Amidohydro_2 não foi relacionado com a decomposição de lignina, similarmente às enzimas da família AA7, estando presentes em hidrolases dependente de metal.

A sequência EGX49222.1 apresentou, além dos domínios FAD_binding_4 e BBE, o domínio CBM_1 com função de interagir com carboidratos. Outras cinco enzimas apresentaram o domínio Chitin_bind_1, sendo que as sequências BAI44126.1 e XP_003717634.1 apresentaram esse domínio duplicado. O domínio Chitin_bind_1 pertence à classe CBM de interação à carboidratos. Os domínios CBM são comuns de serem encontrados em outras enzimas lignolíticas apresentadas no trabalho, estando envolvidos em melhoria da eficiência enzimática.

5.5.10. Resultados dos domínios na família AA8

Todos os domínios encontrados apresentaram o domínio CDH-cyt. Algumas sequências contêmGMC_oxred_N, GMC_oxred_C e CBM_1. O domínio CDH-cyt pode ser encontrado associado apenas com o domínio CBM_1 como no caso da AAU12274.1 e outras enzimas. Não foram encontradas sequências com domínios duplicados.

5.5.11. Resultados dos domínios na família AA9

Todas as sequências apresentaram o domínio Glyco_hydro_61, mas a sequência AEO63926.1 não apresentou em valores superiores ao E-value estipulado. A AEO63926.1 é uma sequência de 80 aminoácidos sendo descrita como *fragment* pelo banco de dados CAZy. Foi encontrado o domínio CBM_1 associado a algumas sequências. O domínio Chitin_bind_1, foi encontrado apenas na sequência ATZ55836.1. Domínios CBM são muito encontrados associados a enzimas envolvidas com metabolismo de carboidratos.

Dois domínios não esperados foram encontrados. O primeiro é o domínio Lactamase_B_2, encontrado na sequência EAA59072.1. Esse domínio está relacionado principalmente com enzimas degradadoras de antibióticos, não sendo encontrado envolvimento direto com a degradação da lignina.

O domínio CFEM foi encontrada na sequência ALL40857.1, descrita em fungos patogênicos. Ela está envolvida em transferência de heme do hospedeiro para o patógeno. Esse domínio é encontrado também em fungos que infectam plantas, podendo estar associado a algum sistema de interação com a lignina, porém nada foi encontrado na literatura (KULKARNI *et al.*, 2003).

As sequências ACE10232.1, ACE10233.1 e ACE10235.1 apresentaram o domínio Glyco_hydro_61 duplicado, tendo tamanho em torno de 195 aminoácidos. Aberrantemente, a sequência EGX48824.1 apresentou 28 domínios CBM_1 e a ACE10235.1 apresenta dois desses domínios. Analisando a ordem de aparecimento dos domínios ACE10235.1, existe um padrão no aparecimento dos domínios Glyco_hydro_61 e CBM_1, o que pode ser um indício que a enzima foi duplicada.

Algumas sequências apresentaram duas regiões de domínio identificadas pelo Pfam, mas com os dois domínios correspondentes com as regiões de alinhamento das demais sequências. Esse fenômeno ocorreu apenas para o domínio Glyco_hydro_61 nas sequências BAV57613.1, CAP64619.1 e VBB76591.1. Os domínios Glyco_hydro_61 apresentaram tamanhos próximos entres as sequências da família AA9, ficando alinhadas na mesma região.

5.5.12. Resultados dos domínios na família AA10

A família AA10 é constituída por poucas sequências de origem fúngica, apresentando apenas 5 sequências. Todas as sequências apresentaram o domínio LPMO_10 com valores de E-value aceitáveis. Todos os domínios apresentaram tamanho de 168 aminoácidos e com mesma região de início e fim de alinhamento. Não foram encontrados domínios duplicados nessas sequências.

5.5.13. Resultados dos domínios na família AA11

O domínio Glyco_hydro_61 foi apresentado por 40 sequências, porém apenas quatro apresentaram com E-value aceitável. A mesma situação aconteceu com o domínio LPMO_10, apresentando 12 sequências com o provável domínio e apenas a sequência ADK37848.1 apresentando o domínio com e-value aceitável. Nenhum outro domínio com e-value aceitável foi registrado, mostrando a inexistência de domínios de referência para a família AA11. A sequência caracterizada no banco de dados Cazy, BAE61530.1, apresenta a estrutura PDB 4MAH e tem o domínio Glyco_hydro_61 com E-value 2.8 e^{-4}, estando próximo ao aceitável e mostrando uma certa semelhança com o domínio presente na família AA9 (HEMSWORTH *et al.*, 2014).

5.5.14. Resultados dos domínios na família AA12

Foi encontrado duas sequências contendo o domínio CBM_1 e quatro contendo o domínio CDH-cyt. Apenas uma sequência foi encontrada contendo ambos os domínios CBM_1 e CDH-cyt. O domínio SGL foi encontrado apenas em uma sequência com E-value satisfatório. Outras 16 sequências apresentaram o domínio SGL com baixo valor estatístico. O domínio GSDH foi encontrado em quatro sequências com E-value satisfatório, que também ocorre em outras quatorze com baixo valor estatístico. Ambos os domínios GSDH e SGL, com baixo valor estatístico podem ser encontrados na mesma sequência, sendo que a ordem de aparecimento e a região dentro da sequência podem variar.Os domínios SGL e GSDH foram estatisticamente significativos e ficaram alinhados juntos. Não foi encontrado sequências com domínios repetidos.

A sequência da estrutura 6H7T não está presente no banco de dados analisados, mas foi relatada recentemente e caracterizada contendo o PQQ interagindo na sua estrutura. Uma análise Pfam da sequência não mostrou domínios com valor estatístico considerável, mas indicouuma baixa associação estatística com o domínio GSDH. O domínio GSDH está presente também na estrutura 1CQ1 da glucose dehydrogenase de *Acinetobactercalcoaceticus*dependente de PQQ, sendo que o domínio GSDH nessa enzima engloba a região de interação com a PQQ.A 1CQ1 é descrita como sendo estruturalmente semelhante a 6H7T, dessa forma, mesmo que o domínio GSDH tenha fraca correlação estatística nas sequências da família AA12, eledeve conter semelhanças com o domínio GSDH (TURBE-DOAN *et al.*, 2019).

A sequência BAP91034.1 é a sequência de aminoácidos de referência no CAZy, mas foi inicialmente referenciada no trabalho de caracterização da família como sequência AB901366. Ela foi a única que apresentou os domínios CBM_1 e CDH-cyt. Ela foi caracterizada no trabalho de Matsumura, que a identificou como pertencente à família AA12 (TURBE-DOAN *et al.*, 2019). Nenhuma sequência duplicada foi encontrada.

5.5.15. Resultados dos domínios na família AA13

Todas as sequências apresentaram o domínio LPMO_10, porém a XP_365988.1 apresentou esse domínio sem valor estatístico aceitável. Foi encontrado também doze sequências com o domínio CBM_20. Muitas das sequências que não apresentaram o domínio CBM_20 apresentaram apenas 250 aminoácidos, porém não foram descritas como *fragment* pelo CAZy. As sequências estão de acordo com o relatado na literatura e pode-se ver uma forte associação com o domínio existente na família AA10. A presença do módulo de ligação

também é bem visível, estando presente em doze das dezenove sequências trabalhadas. Nenhum domínio duplicado foi encontrado (LO LEGGIO *et al.*, 2015). Os domínios LPMO_10 e o domínio CBM_20 ficaram bem alinhados entre as sequências. As sequências CAP61339.1 e VBB78622.1 apresentaram tamanho de domínio LPMO_10 menor das demais e com início de alinhamento tardio.

5.5.16. Resultados dos domínios na família AA14

Apenas três sequências apresentaram domínios caracterizados pelo Pfam. As sequências CDP29754.1, VBB81572.1 e CDZ98532.1 apresentaram o domínio WSC, sendo que a sequência CDZ98532.1 apresentou 5 desses domínios. Nenhum outro domínio foi encontrado com E-value considerável. A sequência AUM86167.1 é caracterizada etem estrutura PDB e não apresentou nenhum tipo de similaridade (mesmo fraca) com outros domínios existentes nas LPMOs de outras famílias.

5.5.17. Resultados dos domínios na família AA16

Foi encontrado vinte e duas sequências com o domínio LPMO_10 com valores de E-value consideráveis e uma sequência com valor fora do esperado. Os domínios identificados tinham tamanhos de 165 aminoácidos, muito próximo ao encontrado para afamília AA10. Em três sequências não foram encontrados o domínio LPMO_10, sendo XP_003190813, CED82808 descrita como *fragment* pelo CAZy, contendo menos de 200 aminoácidos. CED83480.1 é uma sequência de 394 aminoácidos, apresentando tamanho elevado em comparação ao resto do banco. Não foram encontrados domínios duplicados.

Nenhuma referência para domínios CBM foram encontradas nas análises, mesmo a literatura indicando a existência desse tipo de estrutura. Não foram encontrados também relatos de estrutura tridimensional de alguma enzima AA16, porém pode-se notar que existe uma semelhança do domínio descrito com as sequências fúngicas da família AA10. Todos os domínios LPMO_10 tiveram tamanho de sequência e intervalo de alinhamento bem próximos.

5.6. Resultados seleção positiva

Para a análise de seleção positiva, foram baixadas e utilizadas as CDS das sequências proteicas correspondentes. Os números de sequências CDS após a comparação com as sequências proteicas e após a remoção das sequências que não apresentavam um domínio característico, descritos na seção 5.4, estão na Tabela 4.

Tabela 4 - Número de proteínas nas etapas de baixar CDS e verificação dos domínios para entrada do POTION

Família	CDS Baixadas	CDS utilizadas nas análises
AA3sub1	99	90
AA3sub2	359	327
AA3sub3	65	62
AA3sub4	18	-
AA4	25	22
AA5sub1	74	69
AA5sub2	45	-
AA6	161	156
AA7	49	47
AA8	99	-
AA9	415	402
AA10	5	-
AA11	98	-
AA12	37	-
AA13	19	19
AA14	16	-
AA16	25	-

(-) valores que foram utilizadas todas sequências CDS baixadas, pois não apresentaram um domínio característico viável para ser utilizado.

Foram considerados os domínios GMC_oxred_N e GMC_oxred_C para as subfamílias AA3sub1 até AA3sub3. Na família AA3sub4, esses domínios não foram bem característicos da subfamília, sendo utilizadas todas as CDS. Para a família AA4, foram consideradas os domínios FAD_binding_4 e FAD-oxidase_C. Para a família AA5sub1, foi considerado o domínio Glyoxal_oxid_N e DUF1929. Na AA5sub2 foram consideradas todas as CDS. Na família AA6, foi verificado o domínio FMN_red. Na família AA7, foram verificados os domínios FAD_binding_4 e BBE. Para a família AA8, utilizaram-se todas as sequências CDS. Na AA9 utilizou-se o domínio Glyco_hydro_61 para identificar. Na família AA10, foram utilizadas todas as sequências. Na AA13, o domínio LPMO_10 foi considerado o característico. Para as famílias AA11, AA12, AA14 e AA16 foram consideradas todas as sequências.

Na execução do OrthoMCL, foram agrupadas as sequências para serem processadas pelo POTION. O POTION aplica-se vários filtros para a seleção das sequências para evitar falsos positivos e todos esses filtros estão no arquivo de configuração em *conf_potion.conf*(Apêndice E). Foi estabelecido três sequências como o mínimo de sequências no grupo para execução do POTION. Nem todo grupo formado pelo OrthoMCL foi processado pelo POTION, devido ao baixo número de sequências após os filtros. Os grupos também eram avaliados quanto à existência de recombinação entre as sequências, caso fosse encontrada evidências de recombinação, o grupo não era analisado pelo POTION. Em alguns dados processados pelo POTION não era possível serem calculados dados estatísticos e, portanto, essas análises também foram desconsideradas pelo POTION. A tabela 5 mostra os resultados obtidos depois da execução .

Tabela 5 - Número de grupos obtidos pelo OrthoMCL e resultados de seleção neutra e positiva.

Família	Grupos OrthoMCL	NSSP	SP
AA3sub1	8	1	0
AA3sub2	23	4	4
AA3sub3	3	0	0
AA3sub4	1	1	0

AA4	1	0	0
AA5sub1	4	0	1
AA5sub2	5	5	0
AA6	9	1	2
AA7	3	1	0
AA8	11	2	0
AA9	34	11	1
AA10	1	0	0
AA11	7	2	0
AA12	2	0	0
AA13	1	1	0
AA14	1	0	0
AA16	2	0	0

NSSP: Não sofre seleção positiva, esses grupos testados não apresentaram evidências de seleção positiva. SP: Seleção positiva, são os grupos que apresentaram evidências de seleção positiva.

Foi detectada seleção positiva apenas nas famílias AA3sub2, AA5sub1, AA6 e AA9. Em duas famílias foram encontradas evidências de seleção positiva, mas não foi possível identificar o sítio de seleção, isso ocorre pois as técnicas utilizadas pelo POTION concluem que ocorre seleção positiva dentro de um grupo, porém não chegam em um consenso em qual aminoácido isso ocorre, dessa forma, não é possível afirmar com clareza o local que sofre a seleção. A incerteza do sítio de seleção ocorreu na família AA3sub2 nos quatro grupos com evidência de seleção positiva e na família AA6 em apenas um grupo, sendo que no segundo grupo, foi detectada seleção positiva em duas posições. Na família AA5sub1 foi encontrado apenas um grupo contendo um aminoácido positivamente selecionado. Na família AA9 foi encontrada seleção positiva em apenas um grupo que apresentou também seleção em um sítio

aminoacídico. Os arquivos de saída do POTION com a seleção positiva encontram-se no arquivo positive.txt (https://drive.google.com/drive/folders/1_yMvlL_obJqDDQGWEXGga_OeQYkRAC4I).

O primeiro grupo da família AA3sub2 com a numeração AA3sub2_16 apresentou cinco sequências, seguido do grupo AA3sub2_12 com quatro sequências, o grupo AA3sub2_4 com seis e o grupo AA3sub2_18 com três. Todas as sequências nos grupos apresentaram os domínios se alinhando na mesma região e apresentando tamanhos próximos.

O grupo que apresentou seleção positiva na família AA5sub1 foi constituído por três sequências. Todas apresentaram o domínio WSC, sendo que duas apresentaram múltiplos domínios WSC. Os demais domínios tinham tamanhos parecidos e se alinhavam na mesma região. O sítio selecionado positivamente está fora da região identificada como domínio pelo Pfam, utilizando-se para verificação a sequência referência do CEF86606.1 e analisando-se os alinhamentos com as demais sequências.

O grupo AA6_3 foi selecionado positivamente na família AA6, formado por quatro sequências, não teve seu sítio de seleção identificado. O grupo AA6_2 era formado por dezoito sequências. No segundo grupo AA6_2 os domínios apresentaram tamanhos diferentes, sendo a região de início de alinhamento variada e a região do final do alinhamento dos domínios bem próximas. Os sítios selecionados positivamente foram detectados fora do domínio identificado na sequência referência BAK09194.1, usada como molde, porém essa sequência apresentou um tamanho de domínio detectado reduzido em relação às outras do grupo. Na maioria das sequências do grupo AA6_2, essa região está dentro do domínio, podendo ser verificado ao se analisar o alinhamento. Na sequência BAK09194.1, os locais de seleção se alinham com os domínios das demais sequências. Isso mostra que o domínio na sequência BAK09194.1 pode ter sido parcialmente identificado e que os sítios selecionados positivamente no grupo AA6_2 se encontram em uma região de domínio. Os locais de seleção positiva se encontram no aminoácido 33 e 64 da sequência BAK09194.1 com 202 aminoácidos.

O grupo selecionado positivamente na família AA9 foi formado por 8 sequências, sendo que duas sequências apresentaram o domínio CBM_1. Os domínios tinham tamanhos e região de alinhamento bem próximos e o sítio selecionado positivamente estava bem no meio da região identificada como domínio Glyco_hydro_61 pelo Pfam e dentro das regiões de domínio alinhadas das demais sequências. A sequência referência utilizada foi a CCT64153.1 com 339 aminoácidos e o sítio selecionado positivamente se encontrava na posição 173.

5.7. Modelagem das sequências com evidência de seleção positiva

As três famílias nas quais detectou-se seleção positiva, foram utilizadas para construção de modelos estruturais por homologia. Para a enzima da família AA5sub1 obtiveram-se valores de qualidade baixos (GMQE - *Global ModelQualityEstimation*), os quais são calculados relacionando-se a estrutura produzida com o modelo utilizado e QMEAN calculado, utilizando-se diferentes propriedades geométricas, as quais fornecem estimativas de qualidade absoluta global. A enzima AA9 e AA6 obtiveram valores estatísticos GMQE e QMEAN aceitáveis para os modelos. Em questão da cobertura, não se encontrou um bom molde para a família AA5sub1, a qual apresentou região de alinhamento baixo que não permitiu a construção do modelo na região selecionada positivamente.

Na família AA6, o molde utilizado foi a estrutura 5MP4 pertencente a levedura *Saccharomyces cerevisiae*. A 5MP4 é uma BQR caracterizada no banco de dados CAZy na família AA6. A estrutura 5MP4 apresentou 64% de identidade com a sequência BAK09194.1 e permitiu criar um modelo que abrangia praticamente toda a estrutura, exceto o primeiro e os últimos três aminoácidos da sequência.

Na família AA9, o molde utilizado foi a estrutura 5NLT, que apresentou 63% de identidade com a CCT64153.1. A estrutura 5NLT cobriu o aminoácido selecionado positivamente e a região de sítio ativo e interação com substrato, porém, só foi possível criar uma estrutura da posição 21 até a posição 235,do total de 339 aminoácidos. A estrutura 5NLT é caracterizada no banco de dados CAZy, na família AA9, pertencendo ao fungo *Collariellavirescens*.

Para inferir a região de ligação do FMN no modelo da família AA6, utilizou-se o MAFFT, alinhando as sequências proteicas da BAK09194.1 com a quinoneoxidoreductase 3B6I de *Escherichia coli*. De acordo com o trabalho de Koch *et al.* (2017), foi mostrado que a estrutura da enzima 3B6I apresenta 44% de similaridade de sequência, similaridade estrutural e similaridade entre os aminoácidos de interação com a FMN com a enzima 5MP4. A estrutura 5MP4 foi utilizada como molde para construção do modelo, permitindo prever o local de ancoragem do FMN na estrutura (KOCH *et al.*, 2017). Para inferir a região de sítio ativo da sequência proteica CCT64153.1, utilizou-se a estrutura da enzima LPMO 5ACF, pertencente à família AA9 do CAZy e ao fungo *Lentinus similis* (FRANDSEN *et al.*, 2016). As imagens das estruturas utilizadas no trabalho foram criadas utilizando o VMD.

Analisando-se o modelo da AA6, foi possível observar que as regiões selecionadas positivamente se encontram longe da região proposta de interação com a FMN. É possível ver

também que o aminoácido 33 (Valina) se encontra próximo aos aminoácidos da ponta N-terminal, e o aminoácido 64 (Prolina) está entre uma hélice e uma região desordenada (Figura 16). Em relação ao modelo da família AA9 (Figura 17), muitas ressalvas tem que ser tomadas, pois mais de 100 aminoácidos não se encontram na estrutura. Percebe-se que o aminoácido 173 (Alanina) selecionado positivamente, por mais que esteja dentro do domínio, está distante da região de sítio ativo e interação com o substrato. É possível que o aminoácido selecionado se encontre na superfície da enzima e em uma região predita pouco estruturada.

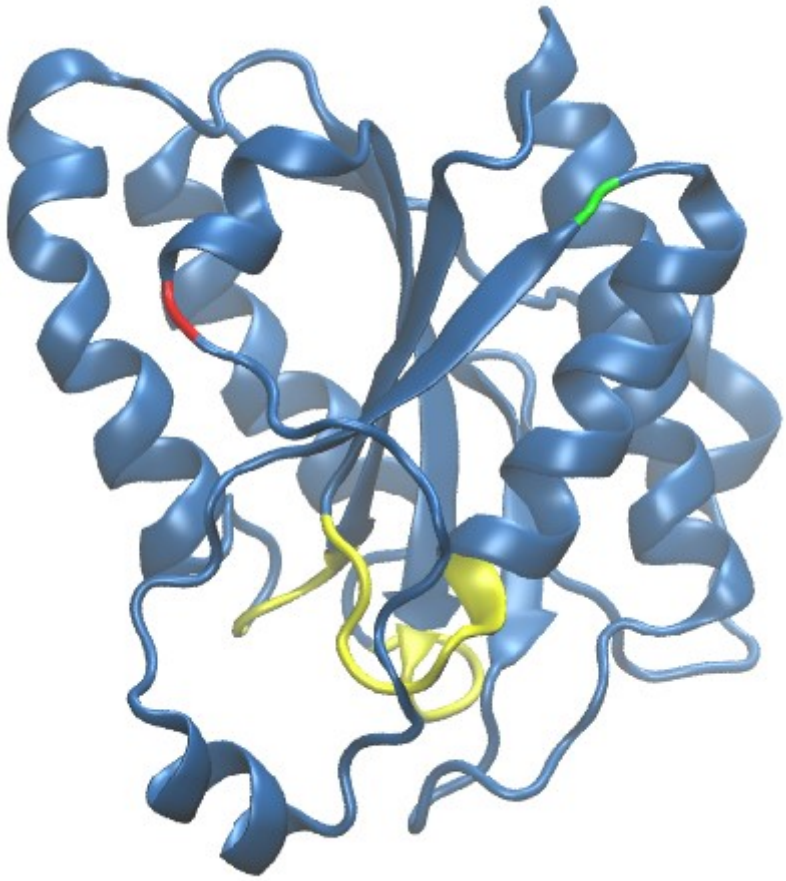

Figura 16 - Estrutura modelada por homologia referente a seleção positiva da família AA6. Em azul se encontra a estrutura proteica modelada da BAK09194.1. Em amarelo se encontram os aminoácidos referentes no sítio de interação com a FMN. Para definir essas posições, utilizou-se como referência a sequência proteica da estrutura PDB 3B6I. Em verde se encontra o resíduo 33 e em vermelho o resíduo 64.

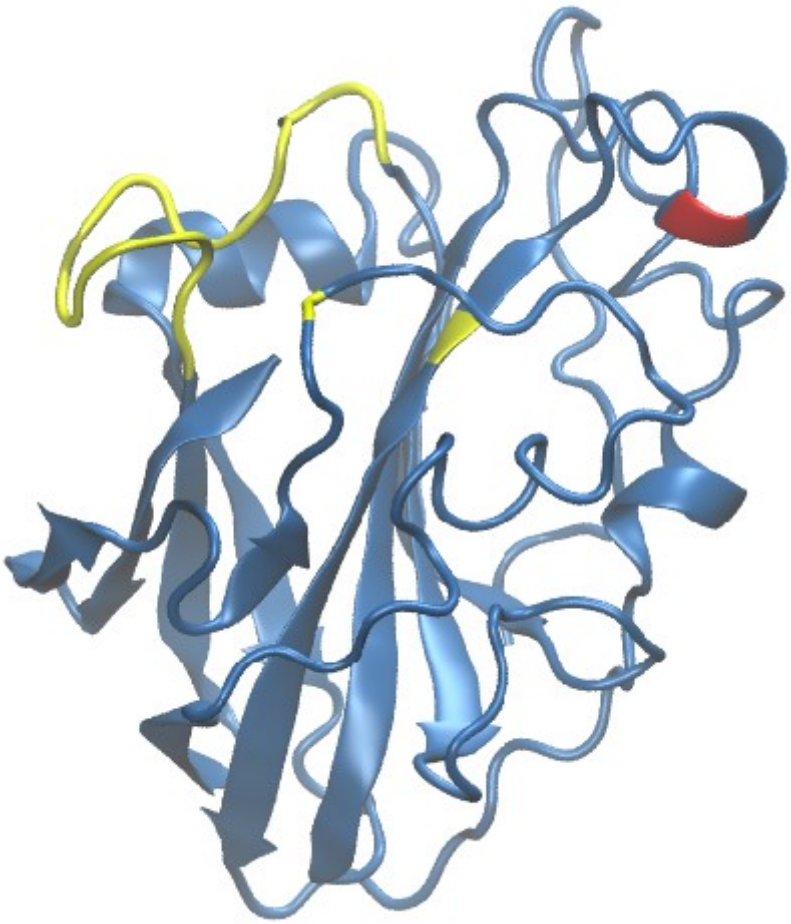

Figura 17 - Estrutura modelada por homologia referente a seleção positiva da família AA9. Em azul, se encontra a estrutura proteica modelada da CCT64153.1. Em amarelo, se encontram os aminoácidos referentes ao sítio ativo e interação ao substrato. Para definir essas posições, utilizou-se como referência a sequência proteica da estrutura PDB 5ACF. Em vermelho, se encontra o resíduo 173.

6. DISCUSSÃO

A degradação da biomassa, considerando todos seus componentes e toda a complexidade da estrutura, é algo complicado e ainda desafiador nos tempos atuais. A construção de um banco interno de sequências baseado nas enzimas indicadas pelo CAZy permitiu que os dados estivessem em um formato acessível para realizar análises. A identificação dos domínios e sua descrição baseada na literatura permitiu gerar dados para que futuros leitores tomem decisões ao utilizar essas sequências em seu trabalho. Um caso desses foi apresentado nesse manuscrito, quando foram usados os domínios descritos para localizar possíveis fragmentos que antes não estavam sendo filtrados na etapa de análise do POTION. A análise dos domínios permitiu a identificação de possíveis mecanismos ainda não descritos pela literatura. Essas identificações ocorreram de forma inesperadas, pois não eram o foco do trabalho encontrar esses indícios, mas serviu para mostrar também que essas enzimas não recebem real atenção de estudo que deveriam.

O tamanho do banco de dados ainda é reduzido, devido ao número limitado de sequências presentes. Porém, podemos considerar a qualidade da curadoria, permitindo informações confiáveis das análises. Esses número reduzidos de sequências dentro de algumas famílias também é devido à recente descoberta de muitas delas, fazendo com que poucas sequências estejam identificadas e anotadas (FILIATRAULT-CHASTEL *et al.*, 2019; TAKEDA *et al.*, 2015). A similaridade entre algumas dessas famílias aumenta a complexidade de entendê-las, necessitando de informações mais complexas para a real identificação (SÜTZL *et al.*, 2019).

6.1. Análise gráfica

Durante a análise da diversidade taxonômica, foi visto que muitos dos grupos apresentaram mais sequências pertencentes ao Filo Ascomycota. Esse número pode ser explicado pelo perfil de estudo dos organismos, um exemplo é o trabalho do Araújo*et al.*, (2018). Neste trabalho é feita uma revisão sobre os genomas sequenciados, sendo encontrada 611 espécies de Ascomycota sequenciadas que cobriam 164 famílias distintas. Nos Ascomycota, ainda foi encontrado um número elevado de múltiplos genomas sequenciados, como em *Saccharomyces cerevisiae*, que apresentava mais do que 100 genomas. Para os Basidiomycota apenas 289 espécies diferentes tinham sido sequenciadas, cobrindo 125 famílias distintas. Deve-se lembrar que é estimado que 98% dos fungos são pertencentes ao

sub-reino Dikarya (Ascomycota + Basidiomycota) e desses é estimado que 64% são Ascomycota (RADEK *et al.*, 2017; STAJICH *et al.*, 2009).

Quanto às análises do tamanho das sequências, pode-se ver uma variação de tamanho dentro das famílias. Isso pode ser explicado por famílias com diferentes domínios constituintes, como na AA3sub1 que apresentou um comportamento bimodal devido a eventos de inexistência de domínios como CDH-cyt e CBM_1, provocando variações de mais de 200 aminoácidos. Eventos com domínios duplicados também podiam ser vistos nos gráficos de tamanho, como no caso da família AA5ub1 que apresentava domínios WSC, podendo chegar até sete repetições, fazendo com que ocorresse um espalhamento dos valores no gráfico. Também foi possível verificar possíveis fragmentos como nas famílias AA4, AA7 e a AA3sub3. Além disso, também foi possível analisar famílias com pouca variação dos domínios e, logo, com pouca variação do tamanho das sequências, como foi o caso da família AA6.

Na família AA5sub2, AA11 e AA14 ocorreram grandes diferenças de tamanho, porém como pouco dados existentes para essas famílias e a inexistência de um domínio característico, impede que seja afirmada a ocorrência de duplicação ou confirme a existência de fragmentos, precisando de mais dados teóricos para reforçar o real motivo do fenômeno. Um exemplo é a família AA11, que tem uma única enzima considerada caracterizada no CAZy, apresentando 421 aminoácidos, sendo que o banco apresenta sequências identificadas com mais do que 900 aminoácidos.

6.2. Domínios

A utilização dos domínios funcionais é uma forma de melhorar a curadoria das sequências identificadas ajudando a localizar sequências que foram selecionadas erroneamente, sequências incompletas ou com domínios duplicados dentro de um banco de dados. A existência de enzimas quiméricas é outro ponto a se analisar. Como pode ser visto nos resultados, vários domínios oriundos de enzimas com funções totalmente distintas foram encontrados juntamente dos domínios comuns às demais sequências do grupo. A existência de enzimas quiméricas funcionais é relatada na literatura como no trabalho de Quin*et al.* (2010), Saadat*et al.* (2017) e no trabalho de Collinet*et al.* (2000). Esse tipo de evento foi encontrado principalmente na subfamília AA3sub2. Essas sequências identificadas como quimeras necessitam de estudos de bancada para averiguar se existe funcionalidade dos domínios extras ou se interfere na atividade originalmente proposta para essa sequência.

Foi possível identificar várias sequências sem um dos domínios identificados como importantes para o funcionamento da enzima, com algumas delas recebendo em sua descrição no CAZy o termo *fragment*. Evidência de fragmentos também podem ser vistas nos gráficos de distribuição de tamanho (Apêndice H), como no caso de subfamílias como a AA3sub1 e AA5sub1 e de famílias como AA4 e AA9.

Foi possível ver que algumas famílias não apresentam domínios característicos e descritos. A falta desses domínios podem ser vistos em famílias como a AA14, a qual não apresentava nenhum domínio característico associado à região catalítica, ou na família AA11, a qual variava a identificação entre os domínios existentes na AA9 e AA10. A identificação de domínios específicos ajudaria na verificação das sequências após a anotação funcional, como também para verificar se as sequências encontradas estão completas. Um exemplo de como a curadoria por domínio pode ser positiva é mostrado na família AA7, na qual a sequência SMQ49385.1 apresentou características muito discrepantes do que foi identificado nas outras sequências, podendo indicar uma sequência erroneamente identificada. Um outro exemplo pode ser visto na família AA9, a qual apresentou a enzima EGX48824.1 contendo 28 domínios CBM_1: um número muito grande de repetições em relação ao número de repetições desse domínio em outras sequências AA. A falta dos domínios específicos mostra o quanto essas enzimas são pouco estudadas e entendidas.

Durante as análises de domínios, foram realizadas descobertas inesperadas, sendo possível localizar alguns mecanismos teóricos. Alguns desses mecanismos foram descritos em algumas das famílias, mas foram encontrados em outras. Muitos desses mecanismos estão envolvidos com interação a carboidrato, como no caso do domínio CBM_1 na família AA7, sendo descrito aumentando a atividade quando adicionados artificialmente na enzima (FOUMANI *et al.*, 2015). O domínio CBM_1 foi encontrado naturalmente na AA7 e na família AA3sub2. Na família AA5sub2 foram localizadas sequências contendo o domínio PAN_1 que pode estar envolvido com um mecanismo ainda não descrito nessa família associado à interação com carboidratos (GONG *et al.*, 2012; LAW *et al.*, 2013). Por fim, foi encontrado na família AA14 o domínio WSC que apresenta atividade de interação com xilano de madeira (OIDE *et al.*, 2019). A família AA14 é descrita na degradação de xilano cristalino. No trabalho de Couturier*et al.* (2018) foi citado um módulo CBM1 que realiza interação com celulose cristalina e que ocorreu diminuição da atividade catalítica. Pode-se pensar que um dos motivos que diminui a atividade da enzima quando apresenta um módulo CBM1 é que dificulta que a enzima entre em contato com o xilano cristalino. No entanto, como a WSC

interage com o xilano ao invés da celulose, isso permitiria que a enzima ficasse próxima do substrato. Não foram encontrados relatos na literatura sobre enzimas LPMOs contendo domínios WSC. Entretanto, esses mecanismos precisam de validação experimental para sua comprovação e os resultados levantados servem de base para as afirmações aqui feitas. Esses domínios podem aumentar a eficiência da enzima e, dessa forma, reduzir a quantidade necessária em uma reação, reduzindo custos de insumos, caso sejam utilizadas em futuros processos enzimáticos.

Outros mecanismos teóricos também foram encontrados de forma inesperada e servem para mostrar o potencial da técnica utilizada. Na subfamília AA3sub2, foi encontrado uma sequência com o domínio HET, o qual pode estar envolvido com mecanismo de inibição de fungos competidores (ESPAGNE *et al.*, 2002). Mesmo sem muitas evidências, o trabalho de Wichmann*et al.* (2008) descreve um domínio semelhante em uma bactéria que pode estar envolvido com vantagens de crescimento, inibindo fungos com o domínio HET incompatível. Na subfamília AA5sub2, um domínio muito presente foi o F5_F8_type_C que é descrito em interação com fosfolipídios (VEERARAGHAVAN *et al.*, 1998). Na família AA9, foi identificado o domínio CFEM descrito como tendo função de aquisição de heme por fungos patogênicos, podendo ser também uma estratégia para fitopatógenos (NASSER *et al.*, 2016). Para ressaltar a possibilidade dessa quimera ser envolvida com patogenicidade do fungo que a contém, a sequência que apresenta esse domínio está associada ao trabalho de análise do secretoma do fungo patogênico *Phakopsorapachyrhizi*, responsável por provocar a doença da ferrugem em soja. Deve-se lembrar que tais mecanismos ainda necessitam de validação experimental, porém muitas das descobertas teóricas são reforçadas pelas informações na literatura.

6.3. Seleção positiva

A identificação de seleção positiva serve como indício da importância do gene para a sobrevivência de um organismo, utilizando da ideia que a fixação de uma variação pode estar relacionada com alguma vantagem evolutiva. A utilização de organismos distantes evolutivamente se mostrou um desafio para esse trabalho, visto que dentre os vários grupos formados durante a etapa do OrthoMCL, poucos conseguiram ser processados pelo POTION, pois muitos eram retidos nos filtros. As mensagens de erro liberadas pelo programa dificultam uma análise manual de cada família e, com o risco de ter muitos erros humanos, mas permite entender os principais problemas encontrados. A recombinação era muito comum, fazendo com que grupos inteiros de sequências fossem removidos durante a etapa de filtragem.

Grupos com recombinação necessitam ser retirados, pois o evento de recombinação dificulta a construção das árvores filogenéticas de forma a descrever corretamente a evolução das proteínas, podendo gerar erros na identificação dos sítios que sofrem seleção positiva (ANISIMOVA *et al.*, 2003). Outros problemas como a baixa semelhança entre as sequências do grupo, diferenças de tamanho devido a existência de enzimas apresentando mais domínios que outras, levava a remoção de sequências individualmente, que fazia com que o números de sequências dentro de um determinado grupo reduzisse muito a ponto de não ser possível de ser analisado. A determinação dos parâmetros buscava reduzir as retiradas das sequências pelos filtros, porém sempre se buscava garantir análises com padrão de filtragens adequados. Esses problemas, inatos das sequências ou da história evolutiva das enzimas, fez com que apenas 31,9% dos grupos formados apresentassem resultados. Mesmo com esses problemas, ainda foram encontradas evidências de seleção positiva em 21,6% dos dados que tiveram resultados. Esse elevado número de seleção positiva mostra que essas sequências têm papel importante na vida dos organismos que as possuem e provavelmente podem ser importantes nos processos em que estão envolvidas, justificando ainda mais os estudos sobre essas famílias ainda negligenciadas.

Dentre os oito grupos que apresentaram seleção positiva, apenas três tiveram o sítio de seleção positiva identificados, sendo que em duas sequências foram encontradas seleção dentro das regiões preditas como domínios. Dentre as três sequências nas quais foram identificados os sítios (aminoácidos) com seleção positiva, em apenas duas foi encontrada estrutura cristalina que apresentasse grau de semelhança suficiente que tornassem aptas a sua utilização na construção de modelos por homologia, sendo que na família AA9 parte da sequência não conseguiu ser predita, devido à baixa similaridade. Dentre os modelos construídos, nenhuma seleção positiva se encontrou muito próxima ao sítio ativo.

A localização da mutação é um fator importante, facilitando com que essa alteração seja conservada. Mutações em superfícies proteicas são mais aceitáveis estruturalmente e tem tendência a desestabilizar menos a estrutura proteica. Dessa forma quanto mais acesso ao solvente um resíduo tem, maior a probabilidade de a mutação nessa posição ser neutra ou aumentar a estabilidade evitando a desnaturação e inativação da enzima por fatores ambientais (TOKURIKI *et al.*, 2007). Na família AA6 os sítios de seleção positiva se encontram próximos da superfície da enzima. Na família AA9 não se pode afirmar com certeza, pois não foi possível montar estrutura em parte da sequência, mas no modelo até então constituído, o sítio de seleção positiva se encontra em região mais externa da estrutura.

A localização do sítio selecionado positivamente serve de indício que a identificação da região não é por acaso, sendo regiões com maior probabilidade de serem locais com alterações que aumentam a estabilidade da estrutura e consequentemente a sua resistência à desnaturação por fatores ambientais. Outro ponto a se notar é o sítio identificado na família AA6 na posição 33 que se encontra perto da região terminal, podendo estar envolvido com interações com a região N-terminal. A quantidade de estudos ainda é pouca, mas alguns autores correlacionam as regiões terminais das enzimas com um aumento de estabilidade devido a uma provável flexibilidade elevada dessa região (BHARDWAJ *et al.*, 2012; JACOB *et al.*, 2007; MAHANTA *et al.*, 2015). Isso serve de indício que a região selecionada positivamente pode estar envolvida com o aumento da estabilidade da estrutura.

Apesar de terem sido analisados poucos grupos de ortólogos devido a fatores que limitam suas análises, foi ainda possível encontrar uma porcentagem elevada de seleção positiva, demonstrando que as famílias analisadas podem ser de grande interesse para a sobrevivência dos organismos que as possuem. Portanto, faz-se necessário e importante o seu melhor entendimento para o desenvolvimento de processos industriais de degradação da lignina e a disponibilização dos carboidratos da estrutura da biomassa na produção de bens de consumo renováveis. Analisando-se as seleções positivas, foi ainda possível estimar sua localização na estrutura tridimensional, sendo que os resultados encontrados são congruentes com a literatura, servindo de indício que essas regiões foram corretamente identificadas nas análises de seleção positiva.

7. CONCLUSÃO

A boa estruturação do banco de dados internos, com a obtenção dos dados de sequência e de taxonomia, foram um passo primordial para permitir as análises, visto que originalmente os dados estavam disponíveis em um formato que inviabilizava o processamento automatizado. A construção de um banco estruturado permitiu a filtragem e a maior clareza das informações, facilitando o serviço de comparação e estudos posteriores.

Com os dados filtrados, foi possível verificar a distribuição taxonômica e perceber uma preferência de estudo com organismos do Filo Ascomycota. Analisando-se os domínios proteicos conservados, foi possível melhorar a acurácia dos bancos, permitindo uma verificação das sequências quanto a possíveis falsos positivos e também a identificação de prováveis fragmentos, melhorando o entendimento dos dados para diversas utilizações futuras como a anotação de genomas ou a busca por novas sequências das enzimas aqui estudadas em bases de dados genômicos e metagenômicos. A análise dos domínios nos permitiu também identificar possíveis estratégias ainda não descritas para as enzimas identificadas e que podem estar relacionadas com a melhora de atividade enzimática, sendo esses dados sustentados pela literatura.

A análise da seleção positiva também mostrou que essas enzimas apresentam evidências de estar sendo fortemente selecionadas positivamente. Foi possível identificar vários grupos com seleção positiva, mesmo com o número pequeno de grupos passíveis de processamento, devido a características inatas desses grupos. As posições selecionadas positivamente foram verificadas nos modelos construídos, e coincidiram com os dados da literatura especializada que mostram que as regiões externas das proteínas são mais passíveis de alteração.

Esse trabalho poderá ajudar no entendimento dessas enzimas tão negligenciadas, mas também envolvidas na degradação da lignina, garantindo dados mais acurados para a utilização em projetos de pesquisa e na construção biotecnológica de enzimas mais eficientes. Esse melhor entendimento ajudará ainda na melhoria de processos que utilizam fontes renováveis de energia para a construção de um mundo mais sustentável e um futuro melhor.

8. PERSPECTIVAS

- Análise de variantes para as enzimas selecionadas positivamente utilizando técnicas de simulação molecular e testes de atividade enzimática, buscando variantes com melhoria de atividade;

- Análise de bancada dos mecanismos teóricos encontrados neste trabalho, a fim de validar o aumento da atividade enzimática na produção de peróxido de hidrogênio;

- Encontrar novas sequências características para os domínios das enzimas, permitindo melhorar a identificação de novas sequências;

- Criação de um pipeline para melhorar a curadoria de outros bancos de sequência utilizando domínios característicos.

REFERÊNCIAS

ACKERLEY, D. F. *et al.*Chromate-ReducingPropertiesofSolubleFlavoproteinsfrom Pseudomonas putidaand Escherichia coli. **Appliedand Environmental Microbiology**, v. 70, n. 2, p. 873–882, 1 fev. 2004.

ADAMS, J.; KELSO, R.; COOLEY, L. The kelchrepeatsuperfamilyofproteins: propellersofcellfunction. **Trends in CellBiology**, v. 10, n. 1, p. 17–24, jan. 2000.

AGARWALA, R. *et al.*DatabaseresourcesoftheNational Center for BiotechnologyInformation. **NucleicAcidsResearch**, v. 46, n. D1, p. D8–D13, 4 jan. 2018.

AGUILETA, G. *et al.*Rapidlyevolving genes in pathogens: Methods for detecting positive selectionandexamplesamong fungi, bacteria, virusesandprotists. **Infection, Geneticsand Evolution**, v. 9, n. 4, p. 656–670, jul. 2009.

AIZAWA, S. *et al.*StructuralBasisofthe γ-Lactone-RingFormation in AscorbicAcidBiosynthesisbytheSenescenceMarker Protein-30/Gluconolactonase. **PLoS ONE**, v. 8, n. 1, p. e53706, 22 jan. 2013.

ALBRECHT, M.; LENGAUER, T. Pyranose oxidase identified as a memberofthe GMC oxidoreductasefamily. **Bioinformatics**, v. 19, n. 10, p. 1216–1220, 1 jul. 2003.

ALONTAGA, A. Y. *et al.* RWD Domain as an E2 (Ubc9)-Interaction Module. **JournalofBiologicalChemistry**, v. 290, n. 27, p. 16550–16559, 3 jul. 2015.

ALTSCHUL, S. F. *et al.* Basic local alignmentsearch tool. **Journalof Molecular Biology**, v. 215, n. 3, p. 403–410, out. 1990.

ANDRADE, S. L. A. *et al.* Crystal StructureoftheNADH:QuinoneOxidoreductaseWrbAfrom Escherichia coli. **JournalofBacteriology**, v. 189, n. 24, p. 9101–9107, 15 dez. 2007.

ANISIMOVA, M.; NIELSEN, R.; YANG, Z. Effectofrecombinationontheaccuracyofthelikelihoodmethod for detecting positive selectionat amino acid sites. **Genetics**, v. 164, n. 3, p. 1229–36, jul. 2003.

ARAUJO, R.; SAMPAIO-MAIA, B. FungalGenomesandGenotyping. p. 37–81.

BANKAR, S. B. *et al.* Glucose oxidase — An overview. **BiotechnologyAdvances**, v. 27, n. 4, p. 489–501, jul. 2009.

BANNWARTH, M. *et al.* Crystal StructureofPyranose 2-Oxidase fromthe White-RotFungusPeniophora sp. † , ‡. **Biochemistry**, v. 43, n. 37, p. 11683–11690, set. 2004.

BATEMAN, A. UniProt: a worldwide hub ofproteinknowledge. **NucleicAcidsResearch**, v.

47, n. D1, p. D506–D515, 8 jan. 2019.

BAZYKIN, G. A. Changingpreferences: deformationof single position amino acid fitness landscapesandevolutionofproteins. **BiologyLetters**, v. 11, n. 10, p. 20150315, 31 out. 2015.

BENGTSSON-PALME, J. *et al.*Strategiesto improve usabilityand preserve accuracy in biologicalsequencedatabases. **PROTEOMICS**, v. 16, n. 18, p. 2454–2460, 1 set. 2016.

BERMAN, H. M. The Protein Data Bank. **NucleicAcidsResearch**, v. 28, n. 1, p. 235–242, 1 jan. 2000.

BHARDWAJ, A. *et al.* EMERGING ROLE OF N- AND C-TERMINAL INTERACTIONS IN STABILIZING (β;/α) 8 FOLD WITH SPECIAL EMPHASIS ON FAMILY 10 XYLANASES. **ComputationalandStructuralBiotechnologyJournal**, v. 2, n. 3, p. e201209014, set. 2012.

BHUIYAN, N. H. *et al.* Gene expressionprofilingandsilencingrevealthatmonolignolbiosynthesis plays a critical role in penetrationdefence in wheatagainstpowderymildewinvasion. **Journalof Experimental Botany**, v. 60, n. 2, p. 509–521, 6 jan. 2009.

BISSARO, B. *et al.*OxidoreductasesandReactiveOxygenSpecies in ConversionofLignocellulosicBiomass. **Microbiologyand Molecular Biology Reviews**, v. 82, n. 4, 26 set. 2018.

BISWAS, S.; AKEY, J. M. Genomic insights into positive selection. **Trends in Genetics**, v. 22, n. 8, p. 437–446, ago. 2006.

BOUCHEROT, A. Cloningandexpressionofthe mouse glomerular podoplanin homologue gp38P. **NephrologyDialysisTransplantation**, v. 17, n. 6, p. 978–984, 1 jun. 2002.

BUCHFINK, B.; XIE, C.; HUSON, D. H. Fast andsensitiveproteinalignmentusing DIAMOND. **NatureMethods**, v. 12, n. 1, p. 59–60, 17 jan. 2015.

BUGG, T. D. H. *et al.*Pathways for degradationoflignin in bacteriaand fungi. **Natural ProductReports**, v. 28, n. 12, p. 1883, 2011.

CAMERON, M. D.; AUST, S. D. KineticsandReactivityoftheFlavinand Heme CofactorsofCellobioseDehydrogenasefromPhanerochaetechrysosporium. **Biochemistry**, v. 39, n. 44, p. 13595–13601, nov. 2000.

CARFI, A. *et al.* The 3-D structureof a zincmetallo-beta-lactamasefrom Bacillus cereusreveals a new typeofproteinfold. **The EMBO journal**, v. 14, n. 20, p. 4914–21, 16 out. 1995.

CHARLESWORTH, B.; CHARLESWORTH, D. Neutral Variation in theContextofSelection. **Molecular Biologyand Evolution**, v. 35, n. 6, p. 1359–1361, 1 jun. 2018.

CHEN, Y. R.; SARKANEN, S.; WANG, Y. Y. Lignin-degradingenzymeactivities. **Methods in Molecular Biology**, v. 908, p. 251–268, 2012.

CHYLENSKI, P. *et al.*LyticPolysaccharideMonooxygenases in EnzymaticProcessingofLignocellulosicBiomass. **ACS Catalysis**, v. 9, n. 6, p. 4970–4991, 7 jun. 2019.

COLLINET, B. *et al.*FunctionallyAcceptedInsertionsofProteinswithinProteinDomains. **JournalofBiologicalChemistry**, v. 275, n. 23, p. 17428–17433, 9 jun. 2000.

COUTURIER, M. *et al.*Lyticxylan oxidases fromwood-decay fungi unlockbiomassdegradation. **NatureChemicalBiology**, v. 14, n. 3, p. 306–310, 29 mar. 2018.

DA SILVA, A. S. *et al.*Sugarcaneand Woody BiomassPretreatments for EthanolProduction. In: **SustainableDegradationofLignocellulosicBiomass - Techniques, ApplicationsandCommercialization**. InTech, 2013.

DANIEL, G. Microviewof Wood underDegradationbyBacteriaand Fungi. p. 34–72.

DANIEL, G. *et al.*CharacteristicsofGloeophyllumtrabeumAlcohol Oxidase, anExtracellularSourceof H2O2 in Brown RotDecayof Wood. **Appliedand Environmental Microbiology**, v. 73, n. 19, p. 6241–6253, 1 out. 2007.

DAOU, M.; FAULDS, C. B. Glyoxal oxidases: theirnatureandproperties. **World JournalofMicrobiologyandBiotechnology**, v. 33, n. 5, p. 87, 7 maio 2017.

DASHTBAN, M. *et al.***Fungalbiodegradationandenzymaticmodificationofligninternationaljournalof Biochemistryand Molecular Biology**e-CenturyPublishing Corporation, , 2010.

DE GONZALO, G. *et al.*Bacterialenzymesinvolved in lignindegradation. **JournalofBiotechnology**, v. 236, p. 110–119, out. 2016.

DEACON, S. E. *et al.*EnhancedFructose Oxidase Activity in a Galactose Oxidase Variant. **ChemBioChem**, v. 5, n. 7, p. 972–979, 5 jul. 2004.

DELLER, S. *et al.*Characterizationof a Thermostable NADPH:FMN OxidoreductasefromtheMesophilicBacterium Bacillus subtilis †. **Biochemistry**, v. 45, n. 23, p. 7083–7091, jun. 2006.

DELLER, S.; MACHEROUX, P.; SOLLNER, S. Flavin-dependentquinonereductases. **Cellularand Molecular Life Sciences**, v. 65, n. 1, p. 141–160, 15 jan. 2008.

DESAI, M. M.; FISHER, D. S. Beneficial Mutation–Selection Balance andtheEffectofLinkageon Positive Selection. **Genetics**, v. 176, n. 3, p. 1759–1798, jul. 2007.

DOYLE, W. A. *et al.*TwoSubstrateInteraction Sites in Lignin Peroxidase Revealedby Site-DirectedMutagenesis †. **Biochemistry**, v. 37, n. 43, p. 15097–15105, out. 1998.

DRENNAN, C. L. *et al.*RefinedstructuresofoxidizedflavodoxinfromAnacystisnidulans 1 1Edited by R. Huber. **Journalof Molecular Biology**, v. 294, n. 3, p. 711–724, dez. 1999.

DUQUE, A. *et al.*SteamExplosion as LignocellulosicBiomassPretreatment. In: **BiomassFractionation Technologies for a LignocellulosicFeedstockBasedBiorefinery**. Elsevier Inc., 2016. p. 349–368.

DUTTA, R. K. *et al.* Comparativeanalysisofthe metal-dependentstructuralandfunctionalpropertiesof mouse andhuman SMP30. **PLOS ONE**, v. 14, n. 6, p. e0218629, 20 jun. 2019.

EDDY, S. R. **HMMER User'sGuide**. 3. ed.

EL-GEBALI, S. *et al.* The Pfamproteinfamiliesdatabase in 2019. **NucleicAcidsResearch**, v. 47, n. D1, p. D427–D432, 8 jan. 2019.

ELLEGREN, H.; GALTIER, N. Determinantsofgeneticdiversity. **Nature Reviews Genetics**, v. 17, n. 7, p. 422–433, 6 jul. 2016.

ENDO, T.; IKEO, K.; GOJOBORI, T. Large-scalesearch for genes onwhich positive selectionmayoperate. **Molecular Biologyand Evolution**, v. 13, n. 5, p. 685–690, 1 maio 1996.

ERIKSSON, K.-E. L.; BLANCHETTE, R. A.; ANDER, P. **Microbial andEnzymaticDegradationof Wood and Wood Components**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990.

ESPAGNE, E. *et al.* HET-E and HET-D belongto a new subfamilyof WD40 proteinsinvolved in vegetativeincompatibilityspecificity in thefungusPodospora anserina. **Genetics**, v. 161, n. 1, p. 71–81, maio 2002.

EWING, T. A.; GYGLI, G.; VAN BERKEL, W. J. H. A single loop isessential for theoctamerizationofvanillylalcohol oxidase. **The FEBS Journal**, v. 283, n. 13, p. 2546–2559, jul. 2016.

FANG, Z.; SMITH, R. L. (EDS.). **ProductionofBiofuelsandChemicalsfromLignin**. Singapore: Springer Singapore, 2016.

FILIATRAULT-CHASTEL, C. *et al.* AA16, a new lyticpolysaccharidemonooxygenasefamilyidentified in fungalsecretomes. **Biotechnology for Biofuels**, v. 12, n. 1, p. 55, 16 dez. 2019.

FINN, R. D. *et al.*Pfam: theproteinfamiliesdatabase. **NucleicAcidsResearch**, v. 42, n. D1, p. D222–D230, jan. 2014.

FIRBANK, S. J. *et al.* Crystal structureofthe precursor of galactose oxidase: Anunusual self-processingenzyme. **ProceedingsoftheNationalAcademyofSciences**, v. 98, n. 23, p. 12932–12937, 6 nov. 2001.

FORCE, A. *et al.*Preservationofduplicate genes bycomplementary, degenerativemutations. **Genetics**, v. 151, n. 4, p. 1531–45, abr. 1999.

FORSBERG, Z. *et al.*Structuralandfunctionalcharacterizationof a conservedpairofbacterialcellulose-oxidizinglyticpolysaccharidemonooxygenases. **ProceedingsoftheNationalAcademyofSciences**, v. 111, n. 23, p. 8446–8451, 10 jun. 2014.

FOUMANI, M. *et al.*EnhancedPolysaccharideBindingandActivityon Linear β-GlucansthroughAdditionofCarbohydrate-Binding Modules toEither Terminus of a Glucooligosaccharide Oxidase. **PLOS ONE**, v. 10, n. 5, p. e0125398, 1 maio 2015.

FOUMANI, M.; VUONG, T. V.; MASTER, E. R. Alteredsubstratespecificityofthegluco-oligosaccharide oxidase fromAcremoniumstrictum. **BiotechnologyandBioengineering**, v. 108, n. 10, p. 2261–2269, out. 2011.

FOWLER, C. A. Characterisationof novel lignocellulosicenzymesfromtheshipwormsymbiontTeredinibacterturnerae. **PhD thesis**, p. 445, 2018.

FRAAIJE, M. W. *et al.*Covalentflavinylationenhancestheoxidativepowerofvanillyl-alcohol oxidase. **Journalof Molecular Catalysis B: Enzymatic**, v. 21, n. 1–2, p. 43–46, jan. 2003.

FRANDSEN, K. E. H. *et al.* The molecular basisofpolysaccharidecleavagebylyticpolysaccharidemonooxygenases. **NatureChemicalBiology**, v. 12, n. 4, p. 298–303, 29 abr. 2016.

FRANDSEN, K. E. H. *et al.* Insights intoanunusualAuxiliaryActivity 9 familymemberlackingthehistidinebracemotifoflyticpolysaccharidemonooxygenases. **JournalofBiologicalChemistry**, v. 294, n. 45, p. 17117–17130, 8 nov. 2019.

FROMMHAGEN, M. *et*

*al.*LyticpolysaccharidemonooxygenasesfromMyceliophthorathermophila C1 differ in substratepreferenceandreducingagentspecificity. **Biotechnology for Biofuels**, v. 9, n. 1, p. 186, 31 dez. 2016.

GHOREISHI, S.; BARTH, T.; HERMUNDSGÅRD, D. H. EffectofReactionConditionsonCatalyticandNoncatalyticLigninSolvolysis in Water Media Investigated for a 5 L Reactor. **ACS Omega**, v. 4, n. 21, p. 19265–19278, 19 nov. 2019.

GIARDINA, P. *et al.*Laccases: a never-endingstory. **Cellularand Molecular Life Sciences**, v. 67, n. 3, p. 369–385, 22 fev. 2010.

GONG, H. *et al.* A Novel PAN/Apple Domain-ContainingProteinfrom Toxoplasma gondii: Characterizationand Receptor Identification. **PLoS ONE**, v. 7, n. 1, p. e30169, 19 jan. 2012.

GOSWAMI, P. *et al.*An overview onalcohol oxidases andtheirpotentialapplications. **AppliedMicrobiologyandBiotechnology**, v. 97, n. 10, p. 4259–4275, 26 maio 2013.

GRABBER, J. H. How Do LigninComposition, Structure, and Cross-LinkingAffectDegradability? A Review ofCell Wall ModelStudies. **Crop Science**, v. 45, n. 3, p. 820–831, maio 2005.

GRANDORI, R. *et al.*BiochemicalCharacterizationofWrbA, FoundingMemberof a New Family ofMultimericFlavodoxin-like Proteins. **JournalofBiologicalChemistry**, v. 273, n. 33, p. 20960–20966, 14 ago. 1998.

GRANDORI, R.; CAREY, J. Six new candidate membersofthe α/β twisted open-sheetfamilydetectedbysequencesimilaritytoflavodoxin. **Protein Science**, v. 3, n. 12, p. 2185–2193, dez. 1994.

GUDDAT, L. W.; BARDWELL, J. C.; MARTIN, J. L. Crystal structuresofreducedandoxidizedDsbA: investigationofdomainmotionandthiolatestabilization. **Structure**, v. 6, n. 6, p. 757–767, jun. 1998.

GUILLÉN, D.; SÁNCHEZ, S.; RODRÍGUEZ-SANOJA, R. Carbohydrate-bindingdomains: multiplicityofbiological roles. **AppliedMicrobiologyandBiotechnology**, v. 85, n. 5, p. 1241–1249, 12 fev. 2010.

GUPTA, V. K. (ED.). **Microbial Enzymes in BioconversionsofBiomass**. Cham: Springer InternationalPublishing, 2016. v. 3

HAGHIGHI MOOD, S. *et al.***Lignocellulosicbiomasstobioethanol, a comprehensive review**

with a focusonpretreatmentRenewableandSustainable Energy Reviews. Elsevier Ltd, , 1 nov. 2013.

HALLBERG, B. M. *et al.* A new scaffold for bindinghaem in thecytochromedomainoftheextracellularflavocytochromecellobiosedehydrogenase. **Structure**, v. 8, n. 1, p. 79–88, jan. 2000.

HANKS, S. K.; HUNTER, T. The eukaryoticproteinkinasesuperfamily: kinase (catalytic) domainstructureandclassification 1. **The FASEB Journal**, v. 9, n. 8, p. 576–596, maio 1995.

HARAYAMA, S.; KOK, M.; NEIDLE, E. L. FunctionalandEvolutionaryRelationshipsAmongDiverseOxygenases. **Annual Review ofMicrobiology**, v. 46, n. 1, p. 565–601, out. 1992.

HARRIS, P. V. *et al.*StimulationofLignocellulosicBiomassHydrolysisbyProteinsofGlycosideHydrolase Family 61: StructureandFunctionof a Large, Enigmatic Family. **Biochemistry**, v. 49, n. 15, p. 3305–3316, 20 abr. 2010.

HASSAN, N. *et al.* Crystal structuresofPhanerochaetechrysosporiumpyranose 2-oxidase suggestthatthe N-terminus acts as a propeptidethatassists in homotetramerassembly. **FEBS Open Bio**, v. 3, n. 1, p. 496–504, 1 jan. 2013.

HEDEGÅRD, E. D.; RYDE, U. Molecular mechanismoflyticpolysaccharidemonooxygenases. **Chemical Science**, v. 9, n. 15, p. 3866–3880, 2018.

HEMSWORTH, G. R. *et al.* Discovery andcharacterizationof a new familyoflyticpolysaccharidemonooxygenases. **NatureChemicalBiology**, v. 10, n. 2, p. 122–126, 22 fev. 2014.

HENRIKSSON, G. *et al.*IscellobiosedehydrogenasefromPhanerochaetechrysosporium a lignindegradingenzyme? **Biochimica et Biophysica Acta (BBA) - ProteinStructureand Molecular Enzymology**, v. 1480, n. 1–2, p. 83–91, jul. 2000.

HENRIKSSON, G.; JOHANSSON, G.; PETTERSSON, G. Iscellobiose oxidase fromPhanerochaetechrysosporium a one-electronreductase? **Biochimica et Biophysica Acta (BBA) - Bioenergetics**, v. 1144, n. 2, p. 184–190, set. 1993.

HENRIKSSON, G.; JOHANSSON, G.; PETTERSSON, G. A critical review ofcellobiosedehydrogenases. **JournalofBiotechnology**, v. 78, n. 2, p. 93–113, mar. 2000.

HERNÁNDEZ-ORTEGA, A.; FERREIRA, P.; MARTÍNEZ, A. T. Fungalaryl-alcohol

oxidase: a peroxide-producingflavoenzymeinvolved in lignindegradation. **AppliedMicrobiologyandBiotechnology**, v. 93, n. 4, p. 1395–1410, 17 fev. 2012.

HERZOG, P. L. *et al.*Versatile Oxidase andDehydrogenaseActivitiesofBacterialPyranose 2-Oxidase Facilitate Redox CyclingwithManganese Peroxidase In Vitro. **Appliedand Environmental Microbiology**, v. 85, n. 13, 26 abr. 2019.

HIGUCHI, T. Ligninbiochemistry: Biosynthesisandbiodegradation. **Wood Science and Technology**, v. 24, n. 1, p. 23–63, mar. 1990.

HMMER. **HMMER**. Disponível em: <hmmer.org>.

HOFRICHTER, M. Review: ligninconversionbymanganese peroxidase (MnP). **Enzymeand Microbial Technology**, v. 30, n. 4, p. 454–466, abr. 2002.

HOLM, L.; SANDER, C. Anevolutionarytreasure: unificationof a broad set ofamidohydrolasesrelatedtourease. **Proteins**, v. 28, n. 1, p. 72–82, maio 1997.

HONGO, J. A. *et al.* POTION: anend-to-end pipeline for positive Darwinianselectiondetection in genome-scale data throughphylogeneticcomparisonofprotein-coding genes. **BMC Genomics**, v. 16, n. 1, p. 567, 1 dez. 2015.

HUANG, C.-H. *et al.* Crystal StructureofGlucooligosaccharide Oxidase fromAcremoniumstrictum. **JournalofBiologicalChemistry**, v. 280, n. 46, p. 38831–38838, 18 nov. 2005.

HUANG, C.-H. *et al.*Functional Roles ofthe 6- S -Cysteinyl, 8α- N1 -Histidyl FAD in Glucooligosaccharide Oxidase fromAcremoniumstrictum. **JournalofBiologicalChemistry**, v. 283, n. 45, p. 30990–30996, 7 nov. 2008.

HUMPHREY, W.; DALKE, A.; SCHULTEN, K. VMD: Visual molecular dynamics. **Journalof Molecular Graphics**, v. 14, n. 1, p. 33–38, fev. 1996.

ISIKGOR, F. H.; BECER, C. R. Lignocellulosicbiomass: a sustainableplatform for theproductionofbio-basedchemicalsandpolymers. **Polymer Chemistry**, v. 6, n. 25, p. 4497–4559, 7 jul. 2015.

ITO, N. *et al.* Crystal Structureof a Free Radical Enzyme, Galactose Oxidase. **Journalof Molecular Biology**, v. 238, n. 5, p. 794–814, maio 1994.

JABRI, E. *et al.* The crystalstructureofureasefromKlebsiellaaerogenes. **Science**, v. 268, n. 5213, p. 998–1004, 19 maio 1995.

JACOB, E.; UNGER, R. A tale oftwotails: why are terminal residuesofproteinsexposed? **Bioinformatics**, v. 23, n. 2, p. e225–e230, 15 jan. 2007.

JANSEN, G. A. Humanphytanoyl-CoAhydroxylase: resolutionofthe gene structureandthe molecular basisofRefsum'sdisease. **Human Molecular Genetics**, v. 9, n. 8, p. 1195–1200, 1 maio 2000.

JENSEN, K. A. *et al.* Biosyntheticpathway for veratrylalcohol in theligninolyticfungusPhanerochaetechrysosporium. **Appliedand Environmental Microbiology**, v. 60, n. 2, p. 709–714, 1 fev. 1994.

JOHANSEN, K. S. LyticPolysaccharideMonooxygenases: The Microbial Power Tool for LignocelluloseDegradation. **Trends in Plant Science**, v. 21, n. 11, p. 926–936, nov. 2016.

KANAGASUNDARAM, V.; SCOPES, R. Isolationandcharacterizationofthe gene encodinggluconolactonasefromZymomonasmobilis. **Biochimica et Biophysica Acta (BBA) - Gene Structureand Expression**, v. 1171, n. 2, p. 198–200, dez. 1992.

KANYONG, P. *et al.*Enzyme-basedamperometric galactose biosensors: a review. **Microchimica Acta**, v. 184, n. 10, p. 3663–3671, 25 out. 2017.

KERSTEN, P.; CULLEN, D. Copper radical oxidases andrelatedextracellularoxidoreductasesofwood-decayAgaricomycetes. **FungalGeneticsandBiology**, v. 72, p. 124–130, nov. 2014.

KOCH, C. *et al.* Crystal StructureofAlcohol Oxidase fromPichia pastoris. **PLOS ONE**, v. 11, n. 2, p. e0149846, 23 fev. 2016.

KOCH, K. *et al.*Structure, biochemicalandkineticpropertiesofrecombinant Pst2p from Saccharomyces cerevisiae , a FMN-dependent NAD(P)H:quinoneoxidoreductase. **Biochimica et Biophysica Acta (BBA) - ProteinsandProteomics**, v. 1865, n. 8, p. 1046–1056, ago. 2017.

KRACHER, D. *et al.*Extracellularelectrontransfer systems fuelcelluloseoxidativedegradation. **Science**, v. 352, n. 6289, p. 1098–1101, 27 maio 2016.

KULKARNI, R. D.; KELKAR, H. S.; DEAN, R. A. Aneight-cysteine-containing CFEM domainuniqueto a groupoffungalmembraneproteins. **Trends in BiochemicalSciences**, v. 28, n. 3, p. 118–121, mar. 2003.

KUTCHAN, T. M.; DITTRICH, H. CharacterizationandMechanismoftheBerberine Bridge Enzyme, a CovalentlyFlavinylated Oxidase ofBenzophenanthridineAlkaloidBiosynthesis in Plants. **JournalofBiologicalChemistry**, v. 270, n. 41, p. 24475–24481, 13 out. 1995.

LAW, R. H.; ABU-SSAYDEH, D.; WHISSTOCK, J. C. New insights

intothestructureandfunctionoftheplasminogen/plasmin system. **CurrentOpinion in StructuralBiology**, v. 23, n. 6, p. 836–841, dez. 2013.

LEFERINK, N. G. H. *et al.* The growing VAO flavoproteinfamily. **ArchivesofBiochemistryandBiophysics**, v. 474, n. 2, p. 292–301, jun. 2008.

LEVASSEUR, A. *et al.*Expansionoftheenzymaticrepertoireoftheb CAZydatabasetointegrateauxiliary redox enzymes. **Biotechnology for Biofuels**, v. 6, n. 1, p. 41, 2013.

LI, F. *et al.* A LyticPolysaccharideMonooxygenasefrom a White-RotFungus Drives theDegradationofLigninby a Versatile Peroxidase. **Appliedand Environmental Microbiology**, v. 85, n. 9, 1 mar. 2019.

LI, J. *et al.*Solventextractionofantioxidantsfromsteamexplodedsugarcane bagasse andenzymaticconvertibilityofthesolidfraction. **Bioresource Technology**, v. 130, p. 8–15, 1 fev. 2013.

LIN, S.-F. *et al.*Purificationandcharacterizationof a novel glucooligosaccharide oxidase fromAcremoniumstrictum T1. **Biochimica et Biophysica Acta (BBA) - ProteinStructureand Molecular Enzymology**, v. 1118, n. 1, p. 41–47, dez. 1991.

LO LEGGIO, L. *et al.*Structureandboostingactivityof a starch-degradinglyticpolysaccharidemonooxygenase. **Nature Communications**, v. 6, n. 1, p. 5961, 22 maio 2015.

LOMBARD, V. *et al.* The carbohydrate-activeenzymesdatabase (CAZy) in 2013. **NucleicAcidsResearch**, v. 42, n. D1, p. D490–D495, jan. 2014.

LOOSE, J. S. M. *et al.*MultipointPrecisionBindingofSubstrateProtectsLyticPolysaccharideMonooxygenases from Self-Destructive Off-Pathway Processes. **Biochemistry**, v. 57, n. 28, p. 4114–4124, 17 jul. 2018.

MACPHERSON, J. M. *et al.* GenomewideSpatialCorrespondenceBetweenNonsynonymousDivergenceand Neutral PolymorphismRevealsExtensiveAdaptation in Drosophila. **Genetics**, v. 177, n. 4, p. 2083–2099, dez. 2007.

MADADI, M.; ABBAS, A. LigninDegradationbyFungalPretreatment: A Review. **JournalofPlantPathology&Microbiology**, v. 08, n. 02, 2017.

MAHANTA, P. *et al.*Modulationof N- to C-terminal interactionsenhancesproteinstability. 12 jan. 2015.

MARTIN, J. L. Thioredoxin —a fold for allreasons. **Structure**, v. 3, n. 3, p. 245–250, mar. 1995.

MARTÍNEZ, Á. T. *et al.***Biodegradationoflignocellulosics: Microbial, chemical, andenzymaticaspectsofthefungalattackoflignin**. InternationalMicrobiology. **Anais**. SociedadEspanola de Microbiologia, 2005

MATSUMURA, H. *et al.* Discovery of a EukaryoticPyrroloquinolineQuinone-DependentOxidoreductaseBelongingto a New AuxiliaryActivity Family in theDatabaseofCarbohydrate-Active Enzymes. **PLoS ONE**, v. 9, n. 8, p. e104851, 14 ago. 2014.

MATTEVI, A. *et al.* Crystal structuresandinhibitorbinding in theoctamericflavoenzymevanillyl-alcohol oxidase: the shape oftheactive-site cavitycontrolssubstratespecificity. **Structure**, v. 5, n. 7, p. 907–920, jul. 1997.

MORENO, A. D. *et al.* A review ofbiologicaldelignificationanddetoxificationmethods for lignocellulosicbioethanolproduction. **Critical Reviews in Biotechnology**, v. 35, n. 3, p. 342–354, 3 jul. 2015.

MORI, T. *et al.*EffectsofHomologous Expression of 1,4-Benzoquinone ReductaseandHomogentisate 1,2-Dioxygenase Genes on Wood Decay in Hyper-Lignin-DegradingFungusPhanerochaetesordida YK-624. **CurrentMicrobiology**, v. 73, n. 4, p. 512–518, 30 out. 2016.

MUIR, P. *et al.* The real costofsequencing: Scalingcomputationtokeep pace with data generation. **GenomeBiology**, v. 17, n. 1, p. 53, 23 mar. 2016.

NAMEKI, N. *et al.*Solutionstructureofthe RWD domainofthe mouse GCN2 protein. **Protein Science**, v. 13, n. 8, p. 2089–2100, ago. 2004.

NASSER, L. *et al.*Structuralbasisofhaem-ironacquisitionbyfungalpathogens. **NatureMicrobiology**, v. 1, n. 11, p. 16156, 12 nov. 2016.

NGUYEN, Q.-T. *et al.*Structure-BasedEngineeringofPhanerochaetechrysosporiumAlcohol Oxidase for EnhancedOxidative Power towardGlycerol. **Biochemistry**, v. 57, n. 43, p. 6209–6218, 30 out. 2018.

OIDE, S. *et al.*Carbohydrate-bindingpropertyof a cellwallintegrityand stress response component (WSC) domainofanalcohol oxidase fromthe rice blastpathogenPyriculariaoryzae. **Enzymeand Microbial Technology**, v. 125, p. 13–20, jun. 2019.

ORTHOMCL. **OrthoMCL**. Disponível em: <https://orthomcl.org/>.

OUBRIE, A. Structureandmechanismofsolublequinoprotein glucose dehydrogenase. **The EMBO Journal**, v. 18, n. 19, p. 5187–5194, 1 out. 1999.

PARIKKA, K.; MASTER, E.; TENKANEN, M. Oxidationwith galactose oxidase: Multifunctionalenzymaticcatalysis. **Journalof Molecular Catalysis B: Enzymatic**, v. 120, p. 47–59, out. 2015.

PEREIRA, L. *et al.* **Isembolismresistance in plantxylemassociatedwithquantityandcharacteristicsoflignin?Trees - StructureandFunction.** Springer Verlag, , 1 abr. 2018.

PÉREZ-BOADA, M. *et al.*Versatile Peroxidase Oxidationof High Redox PotentialAromaticCompounds: Site-directedMutagenesis, SpectroscopicandCrystallographicInvestigationofThreeLong-range ElectronTransferPathways. **Journalof Molecular Biology**, v. 354, n. 2, p. 385–402, nov. 2005.

PERTILE, G. *et al.*Effectofdifferentorganicwasteoncellulose-degradingenzymessecretedbyPetriellasetifera in thepresenceofcellobioseand glucose. **Cellulose**, v. 26, n. 13–14, p. 7905–7922, 25 set. 2019.

PFAM. **PFAM PF07250**. Disponível em: <http://pfam.xfam.org/family/Glyoxal_oxid_N>.

POLIZELI, M. DE L. T. M.; RAI, M. **Fungalenzymes**. CRC Press, 2013.

POLLEGIONI, L.; TONIN, F.; ROSINI, E. Lignin-degradingenzymes. **FEBS Journal**, v. 282, n. 7, p. 1190–1213, 1 abr. 2015.

PONNUSAMY, V. K. *et al.***A review onligninstructure, pretreatments, fermentationreactionsandbiorefinerypotentialBioresource Technology.** Elsevier Ltd, , 1 jan. 2019.

PRABHU, J. *et al.*Functional Expression oftheEctoineHydroxylase Gene (thpD) fromStreptomyceschrysomallus in Halomonaselongata. **Appliedand Environmental Microbiology**, v. 70, n. 5, p. 3130–3132, 1 maio 2004.

PROSITE. **PROSITE**. Disponível em: <https://prosite.expasy.org/cgi-bin/prosite/prosite-search-ac?PDOC00486>.

QIN, B.*et al.*AnUnusualChimericDiterpeneSynthasefromEmericellavariecolorand Its FunctionalConversioninto a SesterterpeneSynthaseby Domain Swapping. **AngewandteChemieInternationalEdition**, v. 55, n. 5, p. 1658–1661, 26 jan. 2016.

RADEK, R. *et al.* Morphologicand molecular data help adoptingtheinsect-pathogenicnephridiophagids (Nephridiophagidae)

amongtheearlydivergingfungallineages, close totheChytridiomycota. **MycoKeys**, v. 25, p. 31–50, 10 jul. 2017.

RADMAN, M.; TADDEI, F.; MATIC, I. Evolution-driving genes. **Research in Microbiology**, v. 151, n. 2, p. 91–95, mar. 2000.

RAGAUSKAS, A. J. *et al.***Ligninvalorization: Improvingligninprocessing in thebiorefineryScience.** American Association for theAdvancementof Science, , 16 maio 2014.

REVUELTA, M. V. *et al.*ExtensiveExpansionof A1 Family Aspartic Proteinases in Fungi RevealedbyEvolutionaryAnalysesof 107 Complete EukaryoticProteomes. **GenomeBiologyand Evolution**, v. 6, n. 6, p. 1480–1494, jun. 2014.

ROZEWICKI, J. *et al.* MAFFT-DASH: integratedproteinsequenceandstructuralalignment. **NucleicAcidsResearch**, 7 maio 2019.

RUIZ-DUEÑAS, F. J.; MARTÍNEZ, Á. T. Microbial degradationoflignin: how a bulkyrecalcitrantpolymerisefficientlyrecycled in natureandhowwecan take advantageofthis. **Microbial Biotechnology**, v. 2, n. 2, p. 164–177, mar. 2009.

SAADAT, F. A review onchimericxylanases: methodsandconditions. **3 Biotech**, v. 7, n. 1, p. 67, 27 maio 2017.

SABBADIN, F. *et al.* Anancientfamilyoflyticpolysaccharidemonooxygenaseswith roles in arthropoddevelopmentandbiomassdigestion. **Nature Communications**, v. 9, n. 1, p. 756, 22 dez. 2018.

SAMEJIMA, M.; ERIKSSON, K.-E. L. A comparisonofthecatalyticpropertiesofcellobiose: quinoneoxidoreductaseandcellobiose oxidase fromPhanerochaetechrysosporium. **EuropeanJournalofBiochemistry**, v. 207, n. 1, p. 103–107, jul. 1992.

SÁNCHEZ, C. Lignocellulosicresidues: Biodegradationandbioconversionby fungi. **BiotechnologyAdvances**, v. 27, n. 2, p. 185–194, mar. 2009.

SBONER, A. *et al.* The real costofsequencing: Higherthanyouthink! **GenomeBiology**, v. 12, n. 8, p. 125, 25 ago. 2011.

SCHEEFF, E. D.; BOURNE, P. E. Structural Evolution oftheProteinKinase–Like Superfamily. **PLoSComputationalBiology**, v. 1, n. 5, p. e49, 2005.

SCHNELLMANN, J. *et al.* The novel lectin-like protein CHB1 isencodedby a chitin-inducibleStreptomycesolivaceoviridis gene andbindsspecificallytocrystalline ?-chitinof fungi andotherorganisms. **Molecular Microbiology**, v. 13, n. 5, p. 807–819, set. 1994.

SCHNOES, A. M. *et al.*AnnotationError in PublicDatabases: Misannotationof Molecular

Function in EnzymeSuperfamilies. **PLoSComputationalBiology**, v. 5, n. 12, p. e1000605, 11 dez. 2009.

SHIMADA, M. *et al.*Biosynthesisofthesecondarymetaboliteveratrylalcohol in relationtolignindegradation in Phanerochaetechrysosporium. **ArchivesofMicrobiology**, v. 129, n. 4, p. 321–324, jun. 1981.

SHIMIZU, M. *et al.*Metabolicregulationatthetricarboxylicacidandglyoxylatecyclesofthelignin-degradingbasidiomycetePhanerochaetechrysosporiumagainstexogenousadditionofvanilli n. **PROTEOMICS**, v. 5, n. 15, p. 3919–3931, out. 2005.

SHOSEYOV, O.; SHANI, Z.; LEVY, I. CarbohydrateBinding Modules: BiochemicalPropertiesand Novel Applications. **Microbiologyand Molecular Biology Reviews**, v. 70, n. 2, p. 283–295, 1 jun. 2006.

SIEBUM, A. *et al.* Galactose oxidase andalcohol oxidase: Scopeandlimitations for theenzymaticsynthesisofaldehydes. **Journalof Molecular Catalysis B: Enzymatic**, v. 41, n. 3–4, p. 141–145, ago. 2006.

SIGOILLOT, J.-C. *et al.*FungalStrategies for LigninDegradation. p. 263–308.

SIMMONS, T. J. *et al.*Structuralandelectronicdeterminantsoflyticpolysaccharidemonooxygenasereactivityon polysaccharidesubstrates. **Nature Communications**, v. 8, n. 1, p. 1064, 20 dez. 2017.

SINGH, J.; SUHAG, M.; DHAKA, A. **Augmenteddigestionoflignocellulosebysteamexplosion, acidandalkalinepretreatmentmethods: A review CarbohydratePolymers.** Elsevier Ltd, , 6 mar. 2015.

SPARLA, F. *et al.*Cloningandheterologousexpressionof NAD(P)H:quinonereductaseofArabidopsisthaliana , a functional homologue of animal DT-diaphorase. **FEBS Letters**, v. 463, n. 3, p. 382–386, 17 dez. 1999.

SRIDHAR, M. Versatile Peroxidases: Super Peroxidases withPotentialBiotechnologicalApplications-A Mini Review. **JournalofDairy, Veterinary& Animal Research**, v. 4, n. 2, 23 dez. 2016.

STAJICH, J. E. *et al.* The Fungi. **CurrentBiology**, v. 19, n. 18, p. R840–R845, set. 2009.

STEVENS, J. C. *et al.*UnderstandingLaccase-IonicLiquidInteractionstowardBiocatalyticLigninConversion in AqueousIonicLiquids. **ACS SustainableChemistryandEngineering**, v. 7, n. 19, p. 15928–15938, 7 out. 2019.

SÜTZL, L. *et al.*Multiplicityofenzymaticfunctions in theCAZy AA3 family.

AppliedMicrobiologyandBiotechnology, v. 102, n. 6, p. 2477–2492, 6 mar. 2018.

SÜTZL, L. *et al.* The GMC superfamilyofoxidoreductasesrevisited: analysisandevolutionoffungal GMC oxidoreductases. **Biotechnology for Biofuels**, v. 12, n. 1, p. 118, 10 dez. 2019.

TAHA, M. *et al.*EnhancedBiologicalStrawSaccharificationThroughCoculturingofLignocellulose-DegradingMicroorganisms. **AppliedBiochemistryandBiotechnology**, v. 175, n. 8, p. 3709–3728, 1 abr. 2015.

TAKEDA, K. *et al.*Characterizationof a Novel PQQ-DependentQuinohemoproteinPyranoseDehydrogenasefromCoprinopsiscinereaClassified intoAuxiliaryActivities Family 12 in Carbohydrate-Active Enzymes. **PLOS ONE**, v. 10, n. 2, p. e0115722, 13 fev. 2015.

TAKEDA, K. *et al.*Fungal PQQ-dependentdehydrogenasesandtheirpotential in biocatalysis. **CurrentOpinion in ChemicalBiology**, v. 49, p. 113–121, abr. 2019.

TAN, T. C. *et al.*StructuralBasis for BindingofFluorinated Glucose and Galactose to Trametes multicolor Pyranose 2-Oxidase VariantswithImproved Galactose Conversion. **PLoS ONE**, v. 9, n. 1, p. e86736, 21 jan. 2014.

TKAC, J. *et al.*Indirectevidenceof direct electron communication betweentheactive site ofgalactose oxidase and a graphiteelectrode. **Bioelectrochemistry**, v. 56, n. 1–2, p. 23–25, maio 2002.

TOKURIKI, N. *et al.* The StabilityEffectsofProteinMutationsAppeartobeUniversallyDistributed. **Journalof Molecular Biology**, v. 369, n. 5, p. 1318–1332, jun. 2007.

TORDAI, H.; BÁNYAI, L.; PATTHY, L. The PAN module: the N-terminal domainsofplasminogenandhepatocytegrowthfactor are homologouswiththeappledomainsoftheprekallikreinfamilyandwith a novel domainfound in numerousnematodeproteins. **FEBS Letters**, v. 461, n. 1–2, p. 63–67, 12 nov. 1999.

TURBE-DOAN, A. *et al.*TrichodermareeseiDehydrogenase, a PyrroloquinolineQuinone-DependentMemberofAuxiliaryActivity Family 12 oftheCarbohydrate-Active EnzymesDatabase: FunctionalandStructuralCharacterization. **Appliedand Environmental Microbiology**, v. 85, n. 24, 11 out. 2019.

UNIPROT. **UNIPROT Q01738**. Disponível em: <https://www.uniprot.org/uniprot/Q01738>.

UNIPROT. **UNIPROT Q9P928**. Disponível em: <https://www.uniprot.org/uniprot/Q9P928>.

UNIPROT. **UNIPROT P13006**. Disponível em: <https://www.uniprot.org/uniprot/P13006>.

URZÚA, U.; KERSTEN, P. J.; VICUÑA, R. Manganese peroxidase-dependentoxidationofglyoxylicandoxalicacidssynthesizedbyCeriporiopsissubvermispora producesextracellularhydrogen peroxide. **Appliedand Environmental Microbiology**, v. 64, n. 1, p. 68–73, 1 jan. 1998.

VANDEN WYMELENBERG, A. *et al.*Structure, Organization, andTranscriptionalRegulationof a Family ofCopper Radical Oxidase Genes in theLignin-DegradingBasidiomyctePhanerochaetechrysosporium. **Appliedand Environmental Microbiology**, v. 72, n. 7, p. 4871–4877, 1 jul. 2006.

VEERARAGHAVAN, S.; BALEJA, J. D.; GILBERT, G. E. Structureandtopographyofthemembrane-binding C2 domainoffactor VIII in thepresenceofdodecylphosphocholinemicelles. **BiochemicalJournal**, v. 332, n. 2, p. 549–555, 1 jun. 1998.

VRIELINK, A.; LLOYD, L. F.; BLOW, D. M. Crystal structureofcholesterol oxidase fromBrevibacteriumsterolicumrefinedat 1.8 Å resolution. **Journalof Molecular Biology**, v. 219, n. 3, p. 533–554, jun. 1991.

WAKABAYASHI, S. *et al.* The amino acidsequenceof a flavodoxinfromtheeukaryoticred alga Chondruscrispus. **BiochemicalJournal**, v. 263, n. 3, p. 981–984, 1 nov. 1989.

WATERHOUSE, A. *et al.* SWISS-MODEL: homologymodellingofproteinstructuresand complexes. **NucleicAcidsResearch**, v. 46, n. W1, p. W296–W303, 2 jul. 2018.

WESTERMARK, U. *et al.*Cellobiose:QuinoneOxidoreductase, a New Wood-degradingEnzymefrom White-rot Fungi. **Acta ChemicaScandinavica**, v. 28b, p. 209–214, 1974.

WESTERMARK, U.; ERIKSSON, K. E. Purificationandpropertiesofcellobiose: quinoneoxidoreductasefromSporotrichumpulverulentum. **Acta chemicaScandinavica. Series B: Organicchemistryandbiochemistry**, v. 29, n. 4, p. 419–24, 1975.

WICHMANN, G. *et al.* A novel gene, phcAfrom Pseudomonas syringaeinducesprogrammedcell death in thefilamentousfungusNeurospora crassa. **Molecular Microbiology**, v. 68, n. 3, p. 672–689, maio 2008.

WINKLER, A. *et al.* A concertedmechanism for berberine bridge enzyme. **NatureChemicalBiology**, v. 4, n. 12, p. 739–741, 26 dez. 2008.

WOOD, J. D.; WOOD, P. M. Evidencethatcellobiose:quinoneoxidoreductasefromPhanerochaetechrysosporiumis a

breakdownproductofcellobiose oxidase. **Biochimica et Biophysica Acta (BBA) - ProteinStructureand Molecular Enzymology**, v. 1119, n. 1, p. 90–96, fev. 1992.

WYMELENBERG, A. VANDEN *et al.*ComputationalanalysisofthePhanerochaetechrysosporium v2.0 genomedatabaseandmassspectrometryidentificationofpeptides in ligninolyticculturesrevealcomplexmixturesofsecretedproteins. **FungalGeneticsandBiology**, v. 43, n. 5, p. 343–356, maio 2006.

YIN, D. *et al.*Structure–functioncharacterizationreveals new catalyticdiversity in the galactose oxidase andglyoxal oxidase family. **Nature Communications**, v. 6, n. 1, p. 10197, 18 dez. 2015.

YOSHIDA, M. *et al.*CharacterizationofCarbohydrate-BindingCytochrome b562 fromthe White-RotFungusPhanerochaetechrysosporium. **Appliedand Environmental Microbiology**, v. 71, n. 8, p. 4548–4555, 1 ago. 2005.

ZÁMOCKÝ, M. *et al.* Ancestral gene fusion in cellobiosedehydrogenasesreflects a specificevolutionof GMC oxidoreductases in fungi. **Gene**, v. 338, n. 1, p. 1–14, ago. 2004.

ZHAO, Q. *et al.* Crystal structureofthe FMN-bindingdomainofhumancytochrome P450 reductaseat 1.93 Å resolution. **Protein Science**, v. 8, n. 2, p. 298–306, 31 dez. 2008.

Apêndice A - Etapas de criação do banco de dados

download_site_cazy.sh

```
#!/bin/bash
sequencia=${1:-16}
for ARQ in $(seq 1 $sequencia)
do
    echo ".....processando AA$ARQ"
    echo "analisando dados"
    FAM="AA"$ARQ"_eukaryota"
    PASTA="AA"$ARQ"_Cazy_eukaryota_python"
    cd /caminho_para_pasta/$PASTA
#acessa a pagina principal para descobrir o numero de paginas para cada familia
    wget "http://www.cazy.org/$FAM.html" -qO $PASTA
    wait
    PAGS=$(cat $PASTA | grep "</a></span><span" | tr "=" \\n | grep "</a></span><span" | cut -d">" -f2 | cut -d"<" -f1)
    if [  -z $PAGS ];
    then
        PAGS=1
    fi
    PAGS=$(echo "$PAGS-1" |bc)
    echo "$PAGS"
#baixando paginas contendo os metadados
    for page in $(seq 0 $PAGS)
    do
        page=$page"00"
        wget "http://www.cazy.org/$FAM.html?debut_TAXO=$page#pagination_TAXO" -qO $page&
    done
    wait
    echo "$PASTA"
    cd ..
done
```

extrair_dados_cazy.py

```
#!/usr/bin/python
import os
importre
for ARQ in range(1,17):
    print '.....processando %d' %(ARQ)
    FAM='AA%d_Cazy_eukaryota' %(ARQ)
    PASTA="AA%d_Cazy_eukaryota_python"%(ARQ)
    print '%s FAM' %(FAM)
    print '%s PASTA' %(PASTA)
    cd='/caminho_para_pasta/%s/%s' %(PASTA, PASTA)
    f=open(cd, 'r')
    line=re.findall(r".*</a></span><spa.*", f.read())
    f.close()
    line="%s" %(line)
    i=""
    ifline != "[]":
            i=line.split("rel='nofollow'>")[1]
            i=i.split("<")[0]
    if i == "":
            i=1
    PAGS=int(i)
    cd='/caminho_para_pasta/%s/dados_%s' %(PASTA, FAM)
    print "%s" %(PAGS)
    dados_cazy=open(cd, 'w+')
    dados_cazy.write("Protein_Name;;EC#;;Organism;;GenBank;;Uniprot;;PDB;;Sub_gru
po;;site\n")
    for page in range (0, PAGS):
            arquivo="%d00" %(page)
            texto="o"
            numero=0
            print "arq%s, texto%s, numero%s" %(arquivo, texto, numero)
            cd='/caminho_para_pasta/%s/%s' %(PASTA, arquivo)
```

```
#abre os arquivos com os codigos do site x000 para leitura
            with open(cd) as file:
#salva em arquivo temporario
                    cd='/caminho_para_pasta/%s/temporario' %(PASTA)
                    temporario=open(cd,'w+')
                    for line in file:
#define quando começa e quando termina as linhas de cada ID
                            linha= re.findall(r"trvalign", line)
                            linha="%s" %(linha)
                            if linha == "['trvalign']":
                                    numero=numero+1
                                    temporario.close()
                                    temporario=open(cd,'w+')
                                    temporario.write("limpo %d \n" %(numero))

#identifica final das linhas referentes ao ID
                            texto=re.findall(r'td id="separateur2" align="center.*', line)
                            temporario.write(line)
                            texto="%s" %(texto)
#caso o fim das linhas do iD for compativelcomeca a pegar dados do ID
                            if 'td id="separateur2" align="center"' in texto:
#fecha e abre arquivo temporario na opcao de leitura
                                    temporario.close()
                                    temporario=open(cd,'r+')
                                    pegar_dados=temporario.read()
#pega os dados referentes a Protein_name
                                    i=""
                                    Protein_Name=re.findall(r'id="separateur2"> .*',
pegar_dados)
                                    Protein_Name="%s" %(Protein_Name)
                                    ifProtein_Name != "[]":
                                            i=Protein_Name.split('> ')[1]
                                            i=i.split("<")[0]
```

```
                                Protein_Name=i
                                if i == "":
                                        Protein_Name="vazio"
#pega os dados referentes a EC#
                                i=""
                                EC=re.findall(r'http://www.enzyme-database.org.*',
pegar_dados)
                                EC="%s" %(EC)
                                if EC != "[]":
                                        i=EC.split('target="_link">')[1]
                                        i=i.split("<")[0]
                                EC=i
                                if EC == "":
                                        EC="vazio"
#pega os dados referentes a Organism
                                i=""
                                Organism=re.findall(r'target="ncbitaxid">.*',
pegar_dados)
                                Organism="%s" %(Organism)
                                ifOrganism != "[]":
                                        i=Organism.split('target="ncbitaxid">')[1]
                                        i=i.split("<")[0]
                                Organism=i
                                ifOrganism == "":
                                        Organism=re.findall(r'http://www.cazy.org/.*',
pegar_dados)
                                        Organism="%s" %(Organism)
                                        ifOrganism != "[]":
                                                i=Organism.split('><b>')[1]
                                                i=i.split("<")[0]
                                Organism=i
                                ifOrganism == "":
                                        Organism="vazio"
```

```
#pega os dados referentes ao GenBank
                                        i=""
                                        GenBank=re.findall(r'ncbi.nlm.nih.gov/entrez.*',
pegar_dados)
                                        GenBank="%s" %(GenBank)
                                        ifGenBank != "[]":
                                                i=GenBank.split('protein&val=')[1]
                                                i=i.split(" ")[0]
                                        GenBank=i
                                        ifGenBank == "":
                                                GenBank="vazio"
#pega os dados referentes ao GenBank

                                        i=""
                                        Uniprot=re.findall(r'uniprot.org/uniprot.*', pegar_dados)
                                        Uniprot="%s" %(Uniprot)
                                        ifUniprot != "[]":
                                                Uniprot=Uniprot.replace(" ","\n")
                                                i=re.findall(r'uniprot.org/uniprot.*"', Uniprot)
                                                i="%s" %(i)
                                                i=i.replace("uniprot.org/uniprot/","")
                                                i=i.replace("","")
                                                i=i.replace("[","")
                                                i=i.replace("]","")
                                                i=i.replace("","")
                                        Uniprot=i
                                        ifUniprot == "":
                                                Uniprot="vazio"
#pega os dados referentes ao GenBank
                                        i=""
                                        PDB=re.findall(r'rcsb.org/pdb/explore.*', pegar_dados)
                                        PDB="%s" %(PDB)
                                        if PDB != "[]":
```

```
                PDB=PDB.replace(" ","\n")
                i=re.findall(r'target=_link>.*</a>', PDB)
                i="%s" %(i)
                i=i.replace("target=_link>","")
                i=i.replace("</a>","")
                i=i.replace("['","")
                i=i.replace("']","")
                i=i.replace("'","")
        PDB=i
        if PDB == "":
                PDB="vazio"
#pega os dados referentes ao Sub_grupo
        i=""
        Sub_grupo=re.findall(r'td                id="separateur2"
align="center">.*', pegar_dados)
        Sub_grupo="%s" %(Sub_grupo)
        ifSub_grupo != "[]":
                i=Sub_grupo.split('td                id="separateur2"
align="center">')[1]
                i=i.split('<')[0]
        Sub_grupo=i
        ifSub_grupo == "":
                Sub_grupo="vazio"
#obtem o site do organismo
        i=""

    site=re.findall(r'http://www.ncbi.nlm.nih.gov/Taxonomy.*', pegar_dados)
        site="%s" %(site)
        if "," in site:
                site=site.split(",")[0]
        if site != "[]":
                i=site.split("'")[1]
                i=i.split("")[0]
```

```
                                        site=i
                                        if site == "":
                                                site=re.findall(r'http://www.cazy.org/.*',
    pegar_dados)
                                                site="%s" %(site)
                                                if "," in site:
                                                        site=site.split(",")[0]
                                                site=site.split("[")[1]
                                                site=site.split("")[0]
                                        if site == "":
                                                site="vazio"
                                        dados_cazy.write("%s;;%s;;%s;;%s;;%s;;%s;;%s;;%s\n"
    %(Protein_Name,EC,Organism,GenBank,Uniprot,PDB,Sub_grupo,site))
                                temporario.close()
    dados_cazy.close()
```

Comando para contagem de linhas

```
catnome_tabela |wc -l
catnome_tabela | awk -F ";;" '{print $7}' | grep "numero_subfamilia" |wc -l
```

baixar_sequencia.sh

```
#!/bin/bash
sequencia=${1:-16}
for ARQ in $(seq 3 $sequencia)
do
#indica o caminho dos arquivos e extrai os dados de ID
        echo ".....processando AA$ARQ"
        echo "analisando dados"
        FAM="AA"$ARQ"_eukaryota_sequencia"
        PASTA="AA"$ARQ"_eukaryota_sequencia_proteica"
        PASTA_acesso="AA"$ARQ"_Cazy_eukaryota_python"
        dados="dados_AA"$ARQ"_Cazy_eukaryota"
        cd /caminho_para_pasta/$PASTA
```

```
        cat /$PASTA_acesso/$dados | awk -F ";;" '{print $4}' | grep -v "vazio" > links.txt
        sed -i '1d' links.txt
        echo $(cat links.txt | wc -l) sequencias serao baixadas ...
#baixando arquivos
        split -dl190 links.txt part
        n_id=0
        echo "baixando do NCBI..."
        total=$(lspart* | wc -l)
        for arquivo in $(lspart* | tr \s \\n)
        do
                n_id=$(echo "$n_id+1" |bc)
                ids=$(cat $arquivo | tr \\n , | cut -d, -f-200)
                list=${ids: :-1}
                wget
"http://www.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=protein&rettype=fasta&id=$list"   -
qOfasta.$arquivo
        done
        wait
        cat fasta.* | grep -vP "^$" > $FAM.fa
        echo "os arquivos foram salvos em $FAM.fa !"
        cd ..

done
```

baixar_taxo.sh

```
    #!/bin/bash
    sequencia=16
    #faz a analise para cada familia
    for ARQ in $(seq 3 $sequencia)
    do
    #salva as variaveis e vai ate as pastas de arquivo
        echo ".....processando AA$ARQ"
        echo "analisando dados"
```

```
PASTA="AA"$ARQ"_eukaryota_taxonomia"
pasta_site="AA"$ARQ"_Cazy_eukaryota_python"
arquivo_site="dados_AA"$ARQ"_Cazy_eukaryota"
cd /caminho_para_pasta/$PASTA
cp /caminho_para_pasta/$pasta_site/$arquivo_site .
dados=$(cat $arquivo_site | awk -F ";;" '{print $8}'| wc -l )
echo "total $dados"
NOME_anterior=nada

for linha in $(seq 2 $dados)
do
#obtem o nome da especie para salvar no arquivo, edita o nome caso preciso para facilitar o processamento
        site_cazy="nada"
        NOME=$(sed -n "$linha p" $arquivo_site | awk -F ";;" '{print $3}' | sed 's/ /_/g' | sed 's/[/]/-/g' | sed 's/[[]//g'| sed 's/[]]//g')
        site=$(sed -n "$linha p" $arquivo_site | awk -F ";;" '{print $8}')
        echo ".....$NOME $site"
        NOME_anterior=$(ls * | grep "$NOME$" )
#faz download do codigo fonte da pagina comparando para evitar multiplos downloads
        if [ "$NOME" != "$NOME_anterior" ]; then
                wget "$site" -O $NOME
                wait
                site_cazy=$(echo "$site" |grep "cazy")
                if [  -n "$site_cazy" ]; then
                        nova_linha=$(cat $NOME | grep 'http://www.ncbi.nlm.nih.gov/Taxonomy/Browser/wwwtax.cgi?id=' | tr " " \\n | grep 'http://www.ncbi.nlm.nih.gov/Taxonomy/Browser/wwwtax.cgi?id=' | cut -d"" -f2)
                        echo "cazy baixando novo site .....$nova_linha"
                        wget "$nova_linha" -O $NOME
                        wait
                fi
        fi
```

```
        done
        cd ..
    done
```

extrair_taxo.py

```
#!/usr/bin/python
import os
importre
num_list = [2,3]

for ARQ in range(1,17):
        print '.....processando %d' %(ARQ)

        PASTA="AA%d_eukaryota_taxonomia"%(ARQ)
        print '%s PASTA' %(PASTA)
#pasta onde fica os taxos
#arquivos com os IDS processados das sequencias
        arquivo_salvar_taxo="AA%d_eukaryota_taxonomia.txt"%(ARQ)
#arquivo a ser gerado com todos os dados
        todos_os_dados="dados_AA%d_Cazy_eukaryota"%(ARQ)
        cd='/home/igor/Downloads/Cazy_certos/cazy_taxonomia_eukaryota/%s/%s'
%(PASTA, arquivo_salvar_taxo)
        salvar_taxo=open(cd,'w+')
        salvar_taxo.write("1.Protein_Name;;2.EC#;;3.Organism;;4.GenBank;;5.Uniprot;;6.PD
B;;7.Sub_grupo;;8.site;;9.superkingdom;;10.kingdom;;11.subkingdom;;12.phylum;;13.subphy
lum;;14.class;;15.order;;16.family;;17.genus;;18.species\n")
        cd='/home/igor/Downloads/Cazy_certos/cazy_taxonomia_eukaryota/%s/%s'
%(PASTA, todos_os_dados)

        with open(cd) as documento:
                for line in documento:
                        line=line.rstrip()
                        dados_do_id=line.split(';;')[2]
```

```
                dados_do_id=dados_do_id.replace(" ","_")
                dados_do_id=dados_do_id.replace("/","-")
                dados_do_id=dados_do_id.replace("[","")
                dados_do_id=dados_do_id.replace("]","")
                ifdados_do_id != "Organism":

        cd='/home/igor/Downloads/Cazy_certos/cazy_taxonomia_eukaryota/%s/%s'
%(PASTA, dados_do_id)
                taxonomia=open(cd, 'r')
                site_taxo=taxonomia.read()

#pega superkingdom
                ifdados_do_id != "Organism":
                        i=""
                        superkingdom=re.findall(r'TITLE="superkingdom".*',
site_taxo)
                        superkingdom="%s" %(superkingdom)
                        superkingdom=superkingdom.replace("/a>","\n")
                        superkingdom=re.findall(r'superkingdom">.*', superkingdom)
                        superkingdom="%s" %(superkingdom)

                        ifsuperkingdom != "[]":
                                i=superkingdom.split(">")[1]
                                i=i.split("<")[0]
                        superkingdom=i
                        ifsuperkingdom == "":
                                superkingdom="vazio"

#pega kingdom
                ifdados_do_id != "Organism":
                        i=""
                        kingdom=re.findall(r'TITLE="kingdom".*', site_taxo)
                        kingdom="%s" %(kingdom)
```

```
                kingdom=kingdom.replace("/a>","\n")
                kingdom=re.findall(r'kingdom">.*', kingdom)
                kingdom="%s" %(kingdom)

                ifkingdom != "[]":
                        i=kingdom.split(">")[1]
                        i=i.split("<")[0]
                kingdom=i
                ifkingdom == "":
                        kingdom="vazio"
#pega subkingdom
        ifdados_do_id != "Organism":
                i=""
                subkingdom=re.findall(r'TITLE="subkingdom".*', site_taxo)
                subkingdom="%s" %(subkingdom)
                subkingdom=subkingdom.replace("/a>","\n")
                subkingdom=re.findall(r'subkingdom">.*', subkingdom)
                subkingdom="%s" %(subkingdom)

                ifsubkingdom != "[]":
                        i=subkingdom.split(">")[1]
                        i=i.split("<")[0]
                subkingdom=i
                ifsubkingdom == "":
                        subkingdom="vazio"
#pega phylum
        ifdados_do_id != "Organism":
                i=""
                phylum=re.findall(r'TITLE="phylum".*', site_taxo)
                phylum="%s" %(phylum)
                phylum=phylum.replace("/a>","\n")
                phylum=re.findall(r'phylum">.*', phylum)
                phylum="%s" %(phylum)
```

```
                ifphylum != "[]":
                        i=phylum.split(">")[1]
                        i=i.split("<")[0]
                phylum=i
                ifphylum == "":
                        phylum="vazio"
#pega subphylum
        ifdados_do_id != "Organism":
                i=""
                subphylum=re.findall(r'TITLE="subphylum".*', site_taxo)
                subphylum="%s" %(subphylum)
                subphylum=subphylum.replace("/a>","\n")
                subphylum=re.findall(r'subphylum">.*', subphylum)
                subphylum="%s" %(subphylum)

                ifsubphylum != "[]":
                        i=subphylum.split(">")[1]
                        i=i.split("<")[0]
                subphylum=i
                ifsubphylum == "":
                        subphylum="vazio"

#pega classe
        ifdados_do_id != "Organism":
                i=""
                classe=re.findall(r'TITLE="class".*', site_taxo)
                classe="%s" %(classe)
                classe=classe.replace("/a>","\n")
                classe=re.findall(r'class">.*', classe)
                classe="%s" %(classe)

                if classe != "[]":
```

```
                    i=classe.split(">")[1]
                    i=i.split("<")[0]
                classe=i
                if classe == "":
                    classe="vazio"

#pega order
            ifdados_do_id != "Organism":
                i=""
                order=re.findall(r'TITLE="order".*', site_taxo)
                order="%s" %(order)
                order=order.replace("/a>","\n")
                order=re.findall(r'order">.*', order)
                order="%s" %(order)

                iforder != "[]":
                    i=order.split(">")[1]
                    i=i.split("<")[0]
                order=i
                iforder == "":
                    order="vazio"
#pega family
            ifdados_do_id != "Organism":
                i=""
                family=re.findall(r'TITLE="family".*', site_taxo)
                family="%s" %(family)
                family=family.replace("/a>","\n")
                family=re.findall(r'family">.*', family)
                family="%s" %(family)

                iffamily != "[]":
                    i=family.split(">")[1]
                    i=i.split("<")[0]
```

```
                family=i
                iffamily == "":
                        family="vazio"
#pega genus
        ifdados_do_id != "Organism":
                i=""
                genus=re.findall(r'TITLE="genus".*', site_taxo)
                genus="%s" %(genus)
                genus=genus.replace("/a>","\n")
                genus=re.findall(r'genus">.*', genus)
                genus="%s" %(genus)

                if genus != "[]":
                        i=genus.split(">")[1]
                        i=i.split("<")[0]
                genus=i
                if genus == "":
                        genus="vazio"
#pega species
        ifdados_do_id != "Organism":
                i=""
                species=re.findall(r'TITLE="species".*', site_taxo)
                species="%s" %(species)
                species=species.replace("/a>","\n")
                species=re.findall(r'species">.*', species)
                species="%s" %(species)

                ifspecies != "[]":
                        i=species.split(">")[1]
                        i=i.split("<")[0]
                species=i
#pega speciescomplex
                ifspecies == "":
```

```
                    species=re.findall(r'speciesgroup">.*', site_taxo)
                    species="%s" %(species)
                    species=species.replace("/a>","\n")
                    species=re.findall(r'speciesgroup">.*', species)
                    species="%s" %(species)
                if ">" in species :
                    i=species.split(">")[1]
                    i=i.split("<")[0]
                species=i
                ifspecies == "":
                    species="vazio"
                salvar_taxo.write("%s;;
%s;;%s;;%s;;%s;;%s;;%s;;%s;;%s;;%s;;%s\n"
%(line,superkingdom,kingdom,subkingdom,phylum,subphylum,classe,order,family,genus,spe
cies))
        salvar_taxo.close()
```

metadados_sequencia.sh

```
#!/bin/bash
sequencia=16
#Faz o processo para cada sequencia
for ARQ in $(seq 3 $sequencia)
do
#gera as variaveis para trabalho
        echo ".....processando AA$ARQ"
        echo "analisando dados"
        PASTA="AA"$ARQ"_eukaryota_taxonomia"
        pasta_site="AA"$ARQ"_Cazy_eukaryota_python"
        arquivo_site="dados_AA"$ARQ"_Cazy_eukaryota"
        sequencias="AA"$ARQ"_eukaryota_sequencia_proteica"
        fa="AA"$ARQ"_eukaryota_sequencia.fa"
        taxo="AA"$ARQ"_eukaryota_taxonomia.txt"
#entra na pasta
        cd /home/igor/Downloads/Cazy_certos/cazy_taxonomia_eukaryota/$PASTA
```

```
#extrai os id para existentes no arquivo taxonomico bruto
	IDS=$(cat /home/igor/Downloads/Cazy_certos/cazy_sequencia_eukaryota_processadas_2/$sequencias/$fa | grep ">" | cut -d " " -f1 | cut -d ">" -f2 | cut -d":" -f1 | cut -d"|" -f2)
	cat /home/igor/Downloads/Cazy_certos/cazy_sequencia_eukaryota_processadas_2/$sequencias/$fa | grep ">" | cut -d " " -f1 | cut -d ">" -f2 | cut -d":" -f1 | cut -d"|" -f2 > "AA"$ARQ"_IDs_sequencias"
#comeca a escrever os arquivos limpos
	echo "Protein_Name;;EC#;;Organism;;GenBank;;Uniprot;;PDB;;Sub_grupo;;site;;NOME;;superkingdom;;kingdom;;subkingdom;;phylum;;subphylum;;class;;order;;family;;genus;;species" > "AA"$ARQ"_taxonomia_sequencias"
	id_sem_duplicatas=$(echo "$IDS"| sort -u)
	contador=1
#Transfere os dados
	for linha in $id_sem_duplicatas
	do
		auxiliar=$(cat "$taxo" | grep "$linha")
		echo "$auxiliar"
		if [  -z "$auxiliar" ];
		then
			auxiliar=sem_sequencia
			echo "$linha"
		fi

		echo "$auxiliar" >> "AA"$ARQ"_taxonomia_sequencias"

	done

	cd ..
done
```

Apêndice B - Etapas de filtragem dos dados da tabela

analise_seq_duplicada.py

```
#!/usr/bin/python3.6
import os
import sys
importsubprocess
fromBio.SeqimportSeq
fromBioimportSeqIO
# ./nome_script.py /caminho/ (o caminho corresponde a pasta com as sequencias fasta que serao comparadas
trabalho = sys.argv
entrada = sys.argv[1]
cd=("seq_repetida.txt")
dados_cazy=open(cd, 'w+')
local=os.system("pwd")
var=subprocess.check_output("ls -1 %s" %(str(entrada)), shell=True)
var=var.rstrip()
var=str(var)
var=var.split("'")[1]
var=var.split("\\n")
#olha se tem repetições com outros grupos
dados_cazy.write("repetições externas")
for i in range(len(var)):
    sub_familia=0
    dados_cazy.write("\n##########################################################################################\n")
    dados_cazy.write("\n.....%s.....\n" %(str(var[i])))
    if "subfamilia" in var[i]:
                                sub_familia=1
    ifsub_familia == 0:
            for j in range(len(var)):
                    if i != j:
```

```
                        dados_cazy.write("\n\n.....%s.....comparando.....%s\n\n"
%(str(var[i]), str(var[j])))
                        for seq_record_1 in SeqIO.parse(entrada+var[i], "fasta"):

                                sub_familia=0

                                if "subfamilia" in var[j]:
                                        sub_familia=1
                                ifsub_familia == 0:
                                        for seq_record_2 in SeqIO.parse(entrada+var[j],
"fasta"):
                                                if seq_record_1.seq == seq_record_2.seq:
                                                        dados_cazy.write("%s %s %s
%s\n" %(seq_record_1.id, seq_record_2.id,str(var[i]),str(var[j])))
#olha se tem repetições internas
dados_cazy.write("\n\n")
dados_cazy.write("\n##########################################################
########################################\n")
dados_cazy.write("\n##########################################################
########################################\n")
dados_cazy.write("\n##########################################################
########################################\n")
dados_cazy.write ("\nrepetições internas\n")
for i in range(len(var)):
    dados_cazy.write("\n.....%s.....\n" %(str(var[i])))
    sub_familia=0
    if "subfamilia" in var[i]:
                        sub_familia=1
    ifsub_familia == 0:
            for seq_record_1 in SeqIO.parse(entrada+var[i], "fasta"):
                    for seq_record_2 in SeqIO.parse(entrada+var[i], "fasta"):
                        if seq_record_1.seq == seq_record_2.seq:
                                if seq_record_1.id != seq_record_2.id:
```

```
                                        dados_cazy.write("%s %s %s %s\n"
%(seq_record_1.id, seq_record_2.id,str(var[i]),str(var[i])))
dados_cazy.close()
```

remove_seq_duplicada.py

```
#!/usr/bin/python3.6
import os
import sys
importsubprocess
fromBio.SeqimportSeq
fromBioimportSeqIO
#formato de entrada script ./remove_seq_duplicada.py
/caminho/AA+_eukaryota_sequencia.fa /caminho/arquivo_taxo
trabalho = sys.argv
entrada = sys.argv[1]
taxo = sys.argv[2]
#abre os arquivos de escrita dos resultados
nome_entrada_salvar=entrada.split("AA")[1]
nome_taxo_salvar=taxo.split("AA")[1]
cd=("seq_sem_repeticao_AA" + nome_entrada_salvar)
print(cd)
dados_seq=open(cd, 'w+')
cd=("taxo_seq_sem_repeticao_AA" + nome_taxo_salvar + ".txt")
print(cd)
dados_taxo=open(cd, 'w+')
dados_taxo.write("Protein_Name;;EC#;;Organism;;GenBank;;Uniprot;;PDB;;Sub_grupo;;
site;;NOME;;superkingdom;;kingdom;;subkingdom;;phylum;;subphylum;;class;;order;;fa
mily;;genus;;species\n")
print("\n########################\n")
#olha se tem repetições internas
##################################################
##################################################
#esse script realiza a retirada da sequencia duplicada e considera de o organismo presente
são os mesmos#
```

```
################################################
################################################
#sequencias que sao repetidas mas precisam ser adicionadas
repetido=[]
#sequencias repetidas que ja tiveram um exemplar adicionado
ignorar=[]
for seq_record_1 in SeqIO.parse(entrada, "fasta"):
    for seq_record_2 in SeqIO.parse(entrada, "fasta"):
            if seq_record_1.seq == seq_record_2.seq and seq_record_1.id != seq_record_2.id:
#adiciona o primeiro elemento seq1 da comparação
                ifstr(seq_record_1.id) in str(repetido) orstr(seq_record_1.id) in str(ignorar):
                        aux=0
                else:
                        repetido.append(seq_record_1.id)
#ve se especie ou genero batem
                with open(taxo) as file:
                        for line in file:
                                line=line.rstrip()
                                linha=line.split(";;")
                                ifstr(seq_record_1.id) in str(linha):
                                        organismo1=line.split(";;")[17]
                                ifstr(seq_record_2.id) in str(linha):
                                        organismo2=line.split(";;")[17]
                ifstr(organismo1) == "vazio" orstr(organismo2) == "vazio":
                        with open(taxo) as file:
                                for line in file:
                                        line=line.rstrip()
                                        linha=line.split(";;")

                                        ifstr(seq_record_1.id) in str(linha):
                                                organismo1=line.split(";;")[16]
```

```
                                        ifstr(seq_record_2.id) in str(linha):
                                            organismo2=line.split(";;")[16]
                    if organismo1 != organismo2:
                        ifstr(seq_record_2.id) in str(repetido) orstr(seq_record_2.id) in
str(ignorar):
                            aux=0
                        else:
                            repetido.append(seq_record_2.id)
                    else:
                        ifstr(seq_record_2.id) in str(repetido) orstr(seq_record_2.id) in
str(ignorar):
                            aux=0
                        else:
                            ignorar.append(seq_record_2.id)
#salvando sequencias que não tem repeticao
seq_colocadas=[]
for seq_record_1 in SeqIO.parse(entrada, "fasta"):
    controle=0
    for seq_record_2 in SeqIO.parse(entrada, "fasta"):
        if seq_record_1.seq == seq_record_2.seq and seq_record_1.id !=
seq_record_2.id:
            controle=1
    for i in range(len(repetido)) :
        if seq_record_1.id == repetido[i]:
            controle=0

    if controle == 0 :
        ignorar_seq=0
        for i in range(len(seq_colocadas)) :
            if seq_record_1.id == seq_colocadas[i]:
                ignorar_seq=1
        ifignorar_seq == 0:
            SeqIO.write(seq_record_1, dados_seq, "fasta")
```

```
#escreve os dados taxonomicos
        with open(taxo) as file:
            for line in file:
                line=line.rstrip()
                linha=line.split(";;")
                ignorar_seq=0
                ifstr(seq_record_1.id) in str(linha):
                    for i in range(len(seq_colocadas)) :
                        if seq_record_1.id == seq_colocadas[i]:
                            ignorar_seq=1
                    ifignorar_seq == 0:
                        dados_taxo.write("%s\n" %(str(line)))
                        seq_colocadas.append(seq_record_1.id)
dados_seq.close()
dados_taxo.close()
```

final_no_rank.py

```
#!/usr/bin/python3.6
import os
importre
import sys
num_list = [2,3]

ARQ="teste"
# AA%d_eukaryota_taxonomia
PASTA= sys.argv[1]
#pasta com os sites baixados
todos_os_dados= sys.argv[2]
#arquivo de taxonomia para obtencao do site
arquivo_salvar_taxo= sys.argv[3]
print ('.....processando %s' %(ARQ))
print ('%s PASTA' %(PASTA))
#arquivo a ser gerado com todos os dados
cd=('%s/%s' %(PASTA, arquivo_salvar_taxo))
```

```
salvar_taxo=open(cd,'w+')
cd=('%s' %(todos_os_dados))
with open(cd) as documento:
    for line in documento:
            line=line.rstrip()
            dados_do_id=line.split(';;')[2]
            dados_do_id=dados_do_id.replace(" ","_")
            dados_do_id=dados_do_id.replace("/","-")
            dados_do_id=dados_do_id.replace("[","")
            dados_do_id=dados_do_id.replace("]","")
            dados_do_id=dados_do_id.replace(";","_")
            dados_do_id=dados_do_id.replace(":","_")
            ifdados_do_id != "Organism":
                    cd=('%s/%s.html' %(PASTA, dados_do_id))
            taxonomia=open(cd, 'r')
            site_taxo=taxonomia.read()
#norank
            classificacao=[]
            nome_classe=[]
            classificacao.append(dados_do_id)
            nome_classe.append(dados_do_id)
            ifdados_do_id != "Organism":
                    ver_linha=site_taxo.split('TITLE')
                    for j in range(len(ver_linha)) :
                            ver_dado=ver_linha[j].split("")
#pega no_rank
                            if  'no rank' in ver_dado:
                                    i=""
                                    no_rank=re.findall(r'="no rank".*', ver_linha[j])
                                    no_rank="%s" %(no_rank)
                                    no_rank=no_rank.replace("/a>","\n")
                                    no_rank=re.findall(r'no rank">.*', no_rank)
                                    no_rank="%s" %(no_rank)
```

```
                    ifno_rank != "[]":
                        i=no_rank.split(">")[1]
                        i=i.split("<")[0]
                    no_rank=i
                    ifno_rank == "":
                        no_rank="vazio"
                    ifno_rank != "vazio":
                        classificacao.append(no_rank)
                        nome_classe.append("no_rank")
#pega        superkingdom
                if  'superkingdom' in ver_dado:
                    i=""
                    superkingdom=re.findall(r'="superkingdom".*',
ver_linha[j])
                    superkingdom="%s" %(superkingdom)
                    superkingdom=superkingdom.replace("/a>","\n")
                    superkingdom=re.findall(r'superkingdom">.*',
superkingdom)
                    superkingdom="%s" %(superkingdom)
                    ifsuperkingdom != "[]":
                        i=superkingdom.split(">")[1]
                        i=i.split("<")[0]
                    superkingdom=i
                    ifsuperkingdom == "":
                        superkingdom="vazio"
                    ifsuperkingdom != "vazio":
                        classificacao.append(superkingdom)
                        nome_classe.append("superkingdom")
#pega kingdom
                if  'kingdom' in ver_dado:
                    i=""
                    kingdom=re.findall(r'="kingdom".*', ver_linha[j])
                    kingdom="%s" %(kingdom)
```

```
                    kingdom=kingdom.replace("/a>","\n")
                    kingdom=re.findall(r'kingdom">.*', kingdom)
                    kingdom="%s" %(kingdom)
                    ifkingdom != "[]":
                        i=kingdom.split(">")[1]
                        i=i.split("<")[0]
                    kingdom=i
                    ifkingdom == "":
                        kingdom="vazio"
                    ifkingdom != "vazio":
                        classificacao.append(kingdom)
                        nome_classe.append("kingdom")
#pega subkingdom
                if  'subkingdom' in ver_dado:
                    i=""
                    subkingdom=re.findall(r'="subkingdom".*',
ver_linha[j])
                    subkingdom="%s" %(subkingdom)
                    subkingdom=subkingdom.replace("/a>","\n")
                    subkingdom=re.findall(r'subkingdom">.*', subkingdom)
                    subkingdom="%s" %(subkingdom)
                    ifsubkingdom != "[]":
                        i=subkingdom.split(">")[1]
                        i=i.split("<")[0]
                    subkingdom=i
                    ifsubkingdom == "":
                        subkingdom="vazio"
                    ifsubkingdom != "vazio":
                        classificacao.append(subkingdom)
                        nome_classe.append("subkingdom")
#pega phylum
                if  'phylum' in ver_dado:
                    i=""
```

```
                    phylum=re.findall(r'="phylum".*', ver_linha[j])
                    phylum="%s" %(phylum)
                    phylum=phylum.replace("/a>","\n")
                    phylum=re.findall(r'phylum">.*', phylum)
                    phylum="%s" %(phylum)
                    ifphylum != "[]":
                            i=phylum.split(">")[1]
                            i=i.split("<")[0]
                    phylum=i
                    ifphylum == "":
                            phylum="vazio"
                    ifphylum != "vazio":
                            classificacao.append(phylum)
                            nome_classe.append("phylum")
#pega subphylum
                if  'subphylum' in ver_dado:
                    i=""
                    subphylum=re.findall(r'="subphylum".*', ver_linha[j])
                    subphylum="%s" %(subphylum)
                    subphylum=subphylum.replace("/a>","\n")
                    subphylum=re.findall(r'subphylum">.*', subphylum)
                    subphylum="%s" %(subphylum)
                    ifsubphylum != "[]":
                            i=subphylum.split(">")[1]
                            i=i.split("<")[0]
                    subphylum=i
                    ifsubphylum == "":
                            subphylum="vazio"
                    ifsubphylum != "vazio":
                            #print( "subphylum %s %s" % (subphylum,
dados_do_id))
                            classificacao.append(subphylum)
                            nome_classe.append("subphylum")
```

```
#pega classe
                    if  'class' in ver_dado:
                        i=""
                        classe=re.findall(r'="class".*', ver_linha[j])
                        classe="%s" %(classe)
                        classe=classe.replace("/a>","\n")
                        classe=re.findall(r'class">.*', classe)
                        classe="%s" %(classe)
                        if classe != "[]":
                            i=classe.split(">")[1]
                            i=i.split("<")[0]
                        classe=i
                        if classe == "":
                            classe="vazio"
                        if classe != "vazio":
                            #print( "class %s %s" % (classe, dados_do_id))
                            classificacao.append(classe)
                            nome_classe.append("classe")
#pega order
                    if  'order' in ver_dado:
                        i=""
                        order=re.findall(r'="order".*', ver_linha[j])
                        order="%s" %(order)
                        order=order.replace("/a>","\n")
                        order=re.findall(r'order">.*', order)
                        order="%s" %(order)
                        iforder != "[]":
                            i=order.split(">")[1]
                            i=i.split("<")[0]
                        order=i
                        iforder == "":
                            order="vazio"
                        iforder != "vazio":
```

```
                    classificacao.append(order)
                    nome_classe.append("order")
#pega family
            if  'family' in ver_dado:
                i=""
                family=re.findall(r'="family".*', ver_linha[j])
                family="%s" %(family)
                family=family.replace("/a>","\n")
                family=re.findall(r'family">.*', family)
                family="%s" %(family)
                iffamily != "[]":
                    i=family.split(">")[1]
                    i=i.split("<")[0]
                family=i
                iffamily == "":
                    family="vazio"
                iffamily != "vazio":
                    classificacao.append(family)
                    nome_classe.append("family")
#pega genus
            if  'genus' in ver_dado:
                i=""
                genus=re.findall(r'="genus".*', ver_linha[j])
                genus="%s" %(genus)
                genus=genus.replace("/a>","\n")
                genus=re.findall(r'genus">.*', genus)
                genus="%s" %(genus)
                if genus != "[]":
                    i=genus.split(">")[1]
                    i=i.split("<")[0]
                genus=i
                if genus == "":
                    genus="vazio"
```

```
                                if genus != "vazio":
                                        classificacao.append(genus)
                                        nome_classe.append("genus")
#pega speciesor pega speciescomplex
                        if  'species' in ver_dadoor 'speciesgroup' in ver_dado :
                                i=""
                                species=re.findall(r'="species".*', ver_linha[j])
                                species="%s" %(species)
                                species=species.replace("/a>","\n")
                                species=re.findall(r'species">.*', species)
                                species="%s" %(species)
                                ifspecies != "[]":
                                        i=species.split(">")[1]
                                        i=i.split("<")[0]
                                species=i
                                ifspecies == "":
                                        species=re.findall(r'speciesgroup">.*',
ver_linha[j])
                                        species="%s" %(species)
                                        species=species.replace("/a>","\n")
                                        species=re.findall(r'speciesgroup">.*', species)
                                        species="complex %s" %(species)
                                if ">" in species :
                                        i=species.split(">")[1]
                                        i=i.split("<")[0]
                                species=i
                                ifspecies == "":
                                        species="vazio"
                                ifspecies != "vazio":
                                        classificacao.append(species)
                                        nome_classe.append("species")
                aux=0
                for k in range(len(classificacao)) :
```

```
                print("%s  ##  %s" %(classificacao[k], nome_classe[k]))
                ifnome_classe[k] == "no_rank":
                        aux=1
                else:
                        aux=0
        salvar_taxo.write("%s;;%s\n" %(aux, dados_do_id))
salvar_taxo.close()
```

apenas_fungo.py

```
#!/usr/bin/python3.6
import os
import sys
importsubprocess
fromBio.SeqimportSeq
fromBioimportSeqIO
#formato de entrada script ./apenas_fungo.py AA+_eukaryota_sequencia.fa
AA+_taxonomia_sequencias.txt
trabalho = sys.argv
entrada = sys.argv[1]
taxo = sys.argv[2]
#abre os arquivos a serem salvos
nome_entrada_salvar=entrada.split("AA")[1]
nome_taxo_salvar=taxo.split("AA")[1]
cd=("seq_sem_repeticao_apenas_fungo_AA" + nome_entrada_salvar)
print(cd)
dados_seq=open(cd, 'w+')
cd=("taxo_sem_repeticao_apenas_fungo_AA" + nome_taxo_salvar)
print(cd)
dados_taxo=open(cd, 'w+')
dados_taxo.write("Protein_Name;;EC#;;Organism;;GenBank;;Uniprot;;PDB;;Sub_grupo;;
site;;superkingdom;;kingdom;;subkingdom;;phylum;;subphylum;;class;;order;;family;;gen
us;;species\n")
print("\n########################\n")
```

```
#analisa o arquivo de taxonomia para ver se é fungo e salva a sequencia e a taxonomia em novos arquivos
for seq_record_1 in SeqIO.parse(entrada, "fasta"):
    with open(taxo) as file:
            for line in file:
                    line=line.rstrip()
                    linha=line.split(";;")[3]
                    ifstr(seq_record_1.id) == str(linha):
                            linha=line.split(";;")[9]
                            ifstr(linha) == str("Fungi"):
                                    dados_taxo.write("%s\n" %(str(line)))
                                    SeqIO.write(seq_record_1, dados_seq, "fasta")
dados_seq.close()
dados_taxo.close()
```

Identificar se algum fungo não tinha classificação no nível de reino

```
for ARQ in $(seq 3 16);do  echo "$ARQ" ; cat AA"$ARQ"_taxonomia_sequencias.txt | awk -F ";;" '{print $3,$9,$10,$11,$12,$13,$14,$15,$16,$17,$18}'|sort -u |grep "Eukaryota vazio";done
```

Identificar quais organismos tinham vazio entre duas classificações taxonômicas

```
for ARQ in $(seq 3 16);do  echo "AA$ARQ" ; cat AA"$ARQ"_taxonomia_sequencias.txt | awk -F ";;" '{print $3,$9,$10,$11,$12,$13,$14,$15,$16,$17,$18}'|sort -u |grep ".* vazio .*";done
```

dividir_subfamilia.py

```
#!/usr/bin/python3.6
import os
import sys
importsubprocess
fromBio.SeqimportSeq
fromBioimportSeqIO
#formato de entrada script  ./dividir_subfamilia.py sequencias/AA+_eukaryota_sequencia.fa /TAXO n_da_familia_para_gerar_nome_de_arquivo  O sinal de + sera aonde serasubstituido pelo valor ARQ
trabalho = sys.argv
```

```
entrada = sys.argv[1]
taxo = sys.argv[2]
familia=sys.argv[3]
n_taxo=[]
#identifica o numero de subfamilias na família
with open(taxo) as file:
        for line in file:
                line=line.rstrip()
                sub_grupo=line.split(";;")[6]
                ifstr(sub_grupo) != "vazio" andstr(sub_grupo) != "Sub_grupo":
                        ifstr(sub_grupo) in str(n_taxo):
                                a=0
                        else:
                                n_taxo.append(sub_grupo)
print(n_taxo)
for i in range(1,len(n_taxo)+1):
#abre os arquivos para salvar os dados
        cd=("sub_familia_completa_seq_sem_repeticao_apenas_fungo_AA%ssub%s_eukaryo
ta_sequencia.fa" %(str(familia), str(i)))
        print(cd)
        dados_seq=open(cd, 'w+')
cd=("sub_familia_completa_taxo_sem_repeticao_apenas_fungo_AA%ssub%s_taxonomia_se
quencias.txt" %(str(familia), str(i)))
        print(cd)
        dados_taxo=open(cd,                                                    'w+')
        dados_taxo.write("Protein_Name;;EC#;;Organism;;GenBank;;Uniprot;;PDB;;Sub_gru
po;;site;;superkingdom;;kingdom;;subkingdom;;phylum;;subphylum;;class;;order;;family;;gen
us;;species\n")
#Faz a separação por sub familia
        for seq_record_1 in SeqIO.parse(entrada, "fasta"):
                with open(taxo) as file:
                        for line in file:
                                line=line.rstrip()
```

```
                    linha=line.split(";;")
                    sub_grupo=line.split(";;")[6]
                    ifstr(seq_record_1.id) in linha andstr(sub_grupo) == str(i):
                        SeqIO.write(seq_record_1, dados_seq, "fasta")
                        dados_taxo.write("%s\n" %(str(line)))
    dados_seq.close()
    dados_taxo.close()
print("ok")
```

Apêndice C - Etapas de construção dos gráficos

isoformas.py

```
#!/usr/bin/python3.6
import os
import sys
importsubprocess
fromBio.SeqimportSeq
fromBioimportSeqIO
#formato de entrada script ./tamanho_sequencias.py AA+_eukaryota_sequencia.fa AA+_taxonomia_sequencias.txt nome_que_se_quer_salvar
trabalho = sys.argv
entrada = sys.argv[1]
taxo = sys.argv[2]
complemento = sys.argv[3]
nome_entrada_salvar=entrada.split("AA")[1]
nome_taxo_salvar=taxo.split("AA")[1]
print("\n#########################\n")

especies=[]
coluna=17
#Obtem as especies existentes no arquivo
with open(taxo) as file:
    for line in file:
            line=line.rstrip()
            linha=line.split(";;")[coluna]
            ifstr(linha) in str(especies):
                    a=0
            else:
                    especies.append([linha])
#adiciona um contador para cada especie
for i in range(len(especies)) :
    especies[i].append(0)
#conta quantas isoformas
```

```
for seq_record_1 in SeqIO.parse(entrada, "fasta"):
    with open(taxo) as file:
            for line in file:
                    line=line.rstrip()
                    linha=line.split(";;")
                    nome_especie=line.split(";;")[coluna]
                    ifstr(seq_record_1.id) in str(linha):
                            for i in range(len(especies)) :
                                    ifstr(especies[i][0]) == str(nome_especie):
                                            especies[i][1]=especies[i][1]+1
#salva na tabela
cd=(complemento)
dados_cazy=open(cd, 'w+')
for i in range(len(especies)) :
    ifespecies[i][1] > 1:
            if "complex" in especies[i][0]:
                    aux=0
            else:
                    dados_cazy.write(especies[i][0] + ";" + str(especies[i][1]) + "\n")
```

tabelando_diversidade.py

```
#!/usr/bin/python3.6
import os
import sys
importre
fromBio.SeqimportSeq
fromBioimportSeqIO
#formato de entrada script  ./tabelando_diversidade.py /entrada /pasta_onde_quer_salvar
/nome_que_se_quer_salvar
trabalho = sys.argv
entrada = sys.argv[1]
saida = sys.argv[2]
complemento= sys.argv[3]
```

```
print ("executado: " + entrada)
print ("salvar: " + complemento)
print ("#####################################################\n\n")
dados1=open(entrada).readlines()
#cria tabela para cada niveltaxonomico
for k in range(8,18):
    x1=[]
    for i in range(len(dados1)):
            if i ==0:
                    linha=dados1[i].rstrip()
                    linha=linha.upper()
                    if "NOME" in linha:
                            titulo=linha.split(';;')[k+1]
                    else:
                            titulo=linha.split(';;')[k]
#faz a contagem de quantas proteinas pertence a aquele taxo
            if i !=0:
                    linha=dados1[i].rstrip()
                    linha=linha.upper()
                    linha=linha.split(';;')[k]
                    aux=0
                    for j in range(len(x1)):
                            if linha in x1[j]:
                                    aux=1
                                    x1[j][1]=int(x1[j][1]) +1
                    ifaux == 0:
                            x1.append([])
                            (x1[len(x1)-1]).append("%s" %str(linha))
                            (x1[len(x1)-1]).append(1)
#salva os niveistaxonomicos
    cd=(saida+titulo+"_"+complemento)
    dados_cazy=open(cd, 'w+')
    for i in range(len(x1)):
```

```
                dados_cazy.write("%s;%s\n" %(x1[i][0],x1[i][1]))
        dados_cazy.close()
```

tamanho_sequencias.py

```
#!/usr/bin/python3.6
import os
import sys
fromBio.SeqimportSeq
fromBioimportSeqIO
#formato de entrada script ./tamanho_sequencias.py sequencias/AA+_eukaryota_sequencia.fa ./teste/ O sinal de + sera aonde serasubstituido pelo valor ARQ
#abre os arquivos
trabalho = sys.argv
entrada = sys.argv[1]
local_salvar=sys.argv[2]
ARQ=sys.argv[3]
salvar_arq_nome= ARQ.split(".")[0]
cd=local_salvar+("tabela_%s.txt" %str(salvar_arq_nome))
dados_cazy=open(cd, 'w+')
dados_cazy.write("ID;;tamanho_seq\n")
#olha o tamanho e salva no arquivo
for seq_record in SeqIO.parse(entrada, "fasta"):
        seq_ref=seq_record.seq
        seq_ref=seq_ref.upper()
        dados_cazy.write("%s;;%s\n" %(seq_record.id, str(len(seq_ref))))
dados_cazy.close()
print (cd)
```

grafico_diversidade.R

```
#Set a pasta de trabalho
setwd(/home/Graficos_finais")
#Cargar a biblioteca
library("ggplot2")
library("stringr")
```

```
#######################################
# PLOT
#######################################
#Carga do dataset
args<- commandArgs(trailing = TRUE)
caminho <- args[1]
taxo <- args[2]
familia<- args[3]
print(caminho)
#data
data <- read.delim(caminho, sep = ';', header=F)
Enzymes = data[,1]
Quantity = data[,2]
data <- data.frame(Enzymes, Quantity)
data <- subset(data, Enzymes!="VAZIO")
print(data)

concatena = paste("TAXONOMIA:", taxo, familia, sep=" ")
salvar = paste(caminho, "png", sep=".")
print(salvar)
ggplot(data = data, aes(x = Enzymes, y = Quantity), ) +
geom_bar(stat = 'identity', width = 0.7, fill = "steelblue") + # cor das barras
geom_text(aes(x = Enzymes,
              y = Quantity + 0.1,
label = paste0(format(Quantity, digits = 2), ")),
hjust = 0,
size = 0, # valor número das barras
          color = rgb(100, 100, 100, maxColorValue = 255), position = position_dodge(width =
1)) +
ggtitle(concatena) +
labs(y = "Número de sequências") +
coord_flip() +
scale_y_continuous(expand = c(0, 0)) +
```

```
theme_minimal() +
theme(axis.text.x = element_text(colour = 'black', size = 1.5, family="Times New Roman"),
axis.text.y = element_text(colour = 'black', size = 2, family="Times New Roman", face="italic"), # eixo Y
axis.title.x = element_text(colour = 'black', size = 2, face="bold", family="Times New Roman"),
axis.title.y = element_blank(),
axis.line.y = element_line(colour = 'black', size = .2),
panel.background = element_rect(colour = NA),
panel.grid.major.y = element_line(colour = NA),
panel.grid.minor.y = element_line(colour = NA),
panel.grid.major = element_line(colour = 'grey85', size = .2),
panel.grid.minor = element_line(colour = 'grey85', size = .2),
        plot.title = element_text(face="bold", family="Times New Roman", hjust = 0.5, colour = 'black', size = 2)) ->data_plot

ggsave(filename = salvar, plot = data_plot, width = 1, height = 1.5, dpi = 500, units = 'in', device = 'png')
```

grafico_isoforma.R

```
#Set a pasta de trabalho
setwd("/home/Graficos_finais")
#Cargar a biblioteca
library("ggplot2")
library("stringr")
#######################################
# PLOT
#######################################
#Carga do dataset
args<- commandArgs(trailing = TRUE)
caminho <- args[1]
taxo <- args[2]
familia<- args[3]
```

```
print(caminho)
#data <-
read.delim("FAMILY_tabelas_sem_repeticao_apenas_fungos_completa_sub_familia_AA4",
sep = ';')
data <- read.delim(caminho, sep = ';', header=F)
Enzymes = data[,1]
Quantity = data[,2]
data <- data.frame(Enzymes, Quantity)
data <- subset(data, Enzymes!="vazio")
print(data)
concatena = paste(familia, taxo, sep=" ")
salvar = paste(caminho, "png", sep=".")
print(salvar)
ggplot(data = data, aes(x = Enzymes, y = Quantity), ) +
geom_bar(stat = 'identity', width = 0.7, fill = "steelblue") + # cor das barras
geom_text(aes(x = Enzymes,
              y = Quantity + 0.1,
label = paste0(format(Quantity, digits = 2), ")),
hjust = 0,
size = 0, # valor número das barras
          color = rgb(100, 100, 100, maxColorValue = 255), position = position_dodge(width =
1)) +
ggtitle(concatena) +
labs(y = "Número de sequências") +
coord_flip() +
scale_y_continuous(expand = c(0, 0)) +
theme_minimal() +
theme(axis.text.x = element_text(colour = 'black', size = 1.5, family="Times New Roman"),
axis.text.y = element_text(colour = 'black', size = 2, family="Times New Roman",
face="italic"), # eixo Y
axis.title.x = element_text(colour = 'black', size = 2, face="bold", family="Times New
Roman"),
axis.title.y = element_blank(),
```

```
axis.line.y = element_line(colour = 'black', size = .2),
panel.background = element_rect(colour = NA),
panel.grid.major.y = element_line(colour = NA),
panel.grid.minor.y = element_line(colour = NA),
panel.grid.major = element_line(colour = 'grey85', size = .2),
panel.grid.minor = element_line(colour = 'grey85', size = .2),
        plot.title = element_text(face="bold", family="Times New Roman", hjust = 0.5, colour = 'black', size = 2)) ->data_plot

ggsave(filename = salvar, plot = data_plot, width = 1, height = 1, dpi = 500, units = 'in', device = 'png')
```

histograma.R

```
#Set a pasta de trabalho
setwd("/home/Graficos_finais")
#Cargar a biblioteca
library("ggplot2")
library("stringr")
library("scales")
#######################################
# PLOT
#######################################
#Carga do dataset
args<- commandArgs(trailing = TRUE)
caminho <- args[1]
familia<- args[2]
print(caminho)
data <- read.delim(caminho, sep = ';', header=F)
Enzymes = data[,1]
Quantity = data[,2]
data <- data.frame(Enzymes, Quantity)
concatena = paste("Tamanho das sequências CAZy:", familia, sep=" ")
salvar = paste(caminho, "png", sep=".")
```

```
#plota o grafico
ggplot(data = data, aes(x = Quantity), ) +
geom_histogram( fill = "steelblue", binwidth=60, col="white" ,size = 0.2) + # cor das barras
ggtitle(concatena) +
labs(x="Tamanho das sequências", y='Número de sequências') +
scale_y_continuous(expand = c(0, 0), breaks= pretty_breaks()) +
scale_x_continuous(breaks= pretty_breaks(4)) +
theme_minimal() +
theme(axis.text.x = element_text(colour = 'black', size = 1.5, family="Times New Roman"),
axis.text.y = element_text(colour = 'black', size = 1.5, family="Times New Roman"), # eixo Y
axis.title.x = element_text(colour = 'black', size = 2, face="bold", family="Times New Roman"),
axis.title.y = element_text(colour = 'black', size = 2, face="bold", family="Times New Roman"),
axis.line.y = element_line(colour = 'black', size = .05),
axis.line.x = element_line(colour = 'black', size = .05),
panel.background = element_rect(colour = NA),
panel.grid.major.x = element_line(colour = NA),
panel.grid.minor.x = element_line(colour = NA),
panel.grid.major = element_line(colour = 'grey85', size = .15),
panel.grid.minor = element_line(colour = 'grey85', size = .15),
      plot.title = element_text(face="bold", family="Times New Roman", hjust = 0.5, colour = 'black', size = 2 )) ->data_plot

ggsave(filename = salvar, plot = data_plot, width = 1, height = 0.7, dpi = 500, units = 'in', device = 'png')
```

Apêndice D - Alinhamento de sequências e análise dos domínios

convert_clustal_fasta.py

```
#!/usr/bin/python3.6
import os
importre
import sys
fromBio.SeqimportSeq
fromBioimportAlignIO
#formato de entrada script /caminho/estrada.clustal /caminho/saida.fasta
entrada = sys.argv[1]
saida = sys.argv[2]
#converte as sequencias
align = AlignIO.read(entrada, "clustal")
AlignIO.write([align], saida, "fasta")
print (entrada)
print (saida)
```

editar_entrada.py

```
#!/usr/bin/python3.6
import os
import sys
importre
fromBio.SeqimportSeq
fromBioimportSeqIO
#formato de entrada script  ./editar_entrada.py /saida_pfam /pasta_onde_quer_salvar
trabalho = sys.argv
entrada = sys.argv[1]
saida = sys.argv[2]
#le a entrada da tabela editando a tabela
tabela_dominio=[]
dados1=open(entrada).readlines()
for j in range(len(dados1)):
    line=dados1[j].rstrip()
    linha_limpa=[]
```

```
    if "#" in line:
            aux=0
    else:
            line=line.rstrip()
            linha=line.split(" ")
            for i in range(len(linha)):
                    if linha[i] != "":
                            linha_limpa.append(linha[i])
            tabela_dominio.append(linha_limpa)
#escreve a tabela editada pronta para ser utilizada
dados_cazy=open(saida, 'w+')
for x in range (len(tabela_dominio)):
    for y in range (len(tabela_dominio[x])):
            if y <(len(tabela_dominio[x])-1):
                    dados_cazy.write("%s;;" %(str(tabela_dominio[x][y])))
            else:
                    dados_cazy.write("%s\n" %(str(tabela_dominio[x][y])))
dados_cazy.close()
```

analises_dominio.py

```
#!/usr/bin/python3.6
import os
import sys
fromBio.SeqimportSeq
fromBioimportSeqIO
fromBioimportAlignIO
#formato de entrada script ./analises_dominio.py pfam_editado.txt
alinhamento.fastasequencias.fastacaminho_pra_salvar/ AA*
trabalho = sys.argv
#arquivo com os dados de dominio
entrada = sys.argv[1]
#arquivo com os dados de alinhamento
alinhamento = sys.argv[2]
#dados das sequencias
```

```
sequencia = sys.argv[3]
#saida do arquivo
saida = sys.argv[4]
#familia
familia = sys.argv[5]
tabela=[]
#garrega a tabela no computador
with open(entrada) as file:
        for line in file:
            line=line.rstrip()
            linha=line.split(";;")
            iffloat(linha[12]) <= float(9.9e-5):
                tabela.append(linha)
#Acha os diferentes dominios existentes no arquivo de dominio que tem e-value determinado anteriormente
dominios=[]
for x in range(len(tabela)):
    aux=0
    if tabela[x][6] in dominios :
        aux=0
    else:
        dominios.append(tabela[x][6])
# captura cada dominio presente em uma sequencia no alinhamento
individual_dominios=[]
for seq_record in AlignIO.read(alinhamento, "fasta"):
    aux_vetor=[]
    aux_vetor.append(seq_record.id)
    for y in range(len(dominios)):
        aux=0
        for z in range(len(tabela)):
            ifstr(seq_record.id) ==str(tabela[z][0])and tabela[z][6] == dominios[y]:
                aux_vetor.append(dominios[y])
                aux=1
```

```
            ifaux == 0 :
                aux_vetor.append("vazio")
    individual_dominios.append(aux_vetor)
#captura quais sequencias presentes no alinhamento n apresentam dominio
n_apresenta_dominio=[]
for x in range(len(dominios)):
    aux_vetor=[]
    aux_vetor.append(dominios[x])
    for y in range(len(individual_dominios)):
            ifstr(dominios[x]) in individual_dominios[y]:
                aux = 0
            else:
                aux_vetor.append(individual_dominios[y][0])

    n_apresenta_dominio.append(aux_vetor)
#captura quais sequencias presentes no alinhamento tem o mesmo dominio duplicado e
quantas vezes
mais_dominio=[]
for x in range(len(dominios)):

    for y in range(len(individual_dominios)):
            aux=0
            for z in range(len(individual_dominios[y])):

                ifstr(dominios[x]) == individual_dominios[y][z]:
                    aux = aux + 1
            ifaux> 1:
                aux_vetor=[]
                aux_vetor.append(dominios[x])
                aux_vetor.append(individual_dominios[y][0])
                aux_vetor.append(aux)
                mais_dominio.append(aux_vetor)
#posicao no alinhamento
```

```
maior=0
menor=100000
posicao_alinhamento=[]
tamanho_dominio_sequencia=[]
for x in range(len(dominios)):
    for y in range(len(tabela)):
            if tabela[y][6] in dominios[x] :
                    for seq_record in AlignIO.read(alinhamento, "fasta"):
                            ifstr(seq_record.id) == str(tabela[y][0]):
                                    aux_vetor=[]
                                    #contador de aa
                                    aux=0
                                    #confirma a chegada de valor de inicio da sequencia no alinhamento
                                    aux_1=0
                                    #conta passos totais
                                    aux_2=0
                                    #confirma a chegada de valor de final da sequencia no alinhamento
                                    aux_4=0
                                    for i in seq_record.seq:
                                            aux_2=aux_2+1
                                            ifstr(i) != "-":
                                                    aux=aux+1
                                            ifint(aux) == int(tabela[y][1]) and aux_1 == 0:
                                                    aux_vetor.append(dominios[x])
                                                    aux_vetor.append(tabela[y][0])
                                                    aux_vetor.append(aux_2)
                                                    aux_1=1
                                            ifint(aux) == int(tabela[y][2]) and aux_4 == 0:
                                                    aux_vetor.append(aux_2)
                                                    aux_4=1
                                    posicao_alinhamento.append(aux_vetor)
```

```
    for y in range(len(tabela)):
        if tabela[y][6] in dominios[x] :
            aux_vetor=[]
            aux_vetor.append(dominios[x])
            aux_vetor.append(tabela[y][0])
            aux_tam= int(tabela[y][2])-int(tabela[y][1]) +1
            aux_vetor.append(aux_tam)
            tamanho_dominio_sequencia.append(aux_vetor)
#acha as posições dos dominios nos alinhamentos, para isso considera as posiçoes dos dominios que se sobrepoem e vai aumentando ate que todos
#os dominios estejam dentro de uma faixa
faixa_posicao_dominio_alinhamento=[]
aux_inicio=0
aux_final=0
for x in range(len(dominios)):
    final= ""
    inicio=""
    aux=0
    aux_vetor=[]
#enqanto todas as sequencias n se encontrarem dentro de uma faixa isso vai se repetindo
    whileaux == 0:
        for y in range(len(posicao_alinhamento)):

            ifstr(posicao_alinhamento[y][0]) == str(dominios[x]) :
                if final == "" orinicio == "" :
                    final=posicao_alinhamento[y][3]
                    inicio =posicao_alinhamento[y][2]
#analisa de a sequenciaesta parcialmente dentro da faixa,exdominio 1-5 faixa4-6 nova faixa 1-6 ou se a faixa pre existente esta dentro do
#dominioexdominio 1-6 faixa 3-5 nova faixa 1-6
                    if (int(posicao_alinhamento[y][2]) >= int(inicio) andint(posicao_alinhamento[y][2]) <= int(final)) or (int(posicao_alinhamento[y][3]) >= int(inicio) andint(posicao_alinhamento[y][3]) <= int(final))
```

```
or(int(posicao_alinhamento[y][2]) <= int(inicio) andint(posicao_alinhamento[y][3]) >= int(final)):

                                        ifint(posicao_alinhamento[y][2]) <int(inicio):
                                                inicio = posicao_alinhamento[y][2]
                                        ifint(posicao_alinhamento[y][3]) >int(final):
                                                final = posicao_alinhamento[y][3]
#adiciona os dados em um vetor
                aux_vetor.append([dominios[x],inicio,final])
                aux=1
#confere se todos estao dentro de uma faixa, caso contrario pega o valor da faixa fora e recomeça uma faixa_2 a partir das posições fora
#da faixa
                for y in range(len(posicao_alinhamento)):
                        aux_1=0
                        ifstr(posicao_alinhamento[y][0]) == str(dominios[x]) :
                                for z in range(len(aux_vetor)):
                                        inicio = aux_vetor[z][1]
                                        final = aux_vetor[z][2]
                                        if (int(posicao_alinhamento[y][2]) >= int(inicio) andint(posicao_alinhamento[y][2]) <= int(final)) or (int(posicao_alinhamento[y][3]) >= int(inicio) andint(posicao_alinhamento[y][3]) <= int(final)):
                                                aux_1=1
#analisa se o dominioesta dentro da faixa, caso contrario pega nova faixa
                                if aux_1 == 0:
                                        aux_inicio = posicao_alinhamento[y][2]
                                        aux_final = posicao_alinhamento[y][3]
                                        aux = 0

                inicio = aux_inicio
                final = aux_final
#adicionas as faixas encontradas para um dominio
    faixa_posicao_dominio_alinhamento.append(aux_vetor)
```

```
tamaho_dominio=[]
for x in range(len(dominios)):
    aux_vetor=[]
    aux_vetor.append(dominios[x])
    for y in range(len(posicao_alinhamento)):
            aux=0
            ifstr(posicao_alinhamento[y][0]) == str(dominios[x]) :

                    aux = int(posicao_alinhamento[y][3]) - int(posicao_alinhamento[y][2])


                    aux_vetor.append(aux)
    tamaho_dominio.append(aux_vetor)
tamanho_faixa=[]
for x in range(len(dominios)):

    for y in range(len(tamaho_dominio)):
            ifstr(tamaho_dominio[y][0]) == str(dominios[x]) :
                    maior_dominio=0
                    menor_dominio=10000000
                    for z in range(1,len(tamaho_dominio[y])):
                            ifint(tamaho_dominio[y][z]) >int( maior_dominio):
                                    maior_dominio=int(tamaho_dominio[y][z])
                            ifint(tamaho_dominio[y][z]) <int( menor_dominio):
                                    menor_dominio=int(tamaho_dominio[y][z])
    for y in range(len(faixa_posicao_dominio_alinhamento)):
            for z in range(len(faixa_posicao_dominio_alinhamento[y])):
                    ifstr(faixa_posicao_dominio_alinhamento[y][z][0]) == str(dominios[x])
:
                            aux=0
                            aux_1=0
                            aux =   int(faixa_posicao_dominio_alinhamento[y][z][2])        -
int(faixa_posicao_dominio_alinhamento[y][z][1])
```

```
                        aux_1 = aux/maior_dominio
                        aux_2 = aux/menor_dominio
                        faixa_posicao_dominio_alinhamento[y][z].append(aux)

    faixa_posicao_dominio_alinhamento[y][z].append(maior_dominio)
                        faixa_posicao_dominio_alinhamento[y][z].append(aux_1)

    faixa_posicao_dominio_alinhamento[y][z].append(menor_dominio)
                        faixa_posicao_dominio_alinhamento[y][z].append(aux_2)
#salvando em arquivos organizados

cd=(saida + "resumo_" + familia)
dados_resumo=open(cd, 'w+')
dados_resumo.write("\n -=-=-=-=-=--=-=--=-=-=-=-=-=-=-=- Dominios totais encontrados -=-=-=-=-=--=-=--=-=-=-=-=-=-=-=- \n")
dados_resumo.write("aqui estao todos dominios presentes nas sequencias que satisfizeram e value\n\n")
for x in range(len(dominios)):
    dados_resumo.write("%s\n" %(str(dominios[x])))

dados_resumo.write("\n -=-=-=-=-=--=-=--=-=-=-=-=-=-=-=- Sequencias que não apresentaram Dominios -=-=-=-=-=--=-=--=-=-=-=-=-=-=-=- \n")
dados_resumo.write("aqui estao todas sequencias que não apresentaram um certo dominio\n\n")
for x in range(len(n_apresenta_dominio)):
    dados_resumo.write("\n********** Dominio %s **********\n\n" %(str(n_apresenta_dominio[x][0])))
    for y in range(1,len(n_apresenta_dominio[x])):
            for seq_record in SeqIO.parse(sequencia, "fasta"):
                    ifstr(n_apresenta_dominio[x][y]) in str(seq_record.id):
                        seq_ref=seq_record.seq
                        seq_ref=seq_ref.upper()
```

```
                    dados_resumo.write("sequencia %s de tamanho %d nao apresentou dominio %s\n\n" %(str(n_apresenta_dominio[x][y]), len(seq_ref), str(n_apresenta_dominio[x][0])))



dados_resumo.write("\n -=-=-=-=-=--=-=--=-=-=-=-=-=-=-=- Sequencias com dominios duplicados -=-=-=-=-=--=-=--=-=-=-=-=-=-=-=- \n")
dados_resumo.write("aqui estao todas sequencias que apresentaram dominios duplicados\n\n")
for x in range(len(dominios)):
    dados_resumo.write("\n********************** Dominio %s **********************\n\n" %(str(dominios[x])))
    for y in range(len(mais_dominio)):
            ifstr(mais_dominio[y][0]) == str(dominios[x]) :
                    dados_resumo.write("sequencia %s apresentou %s dominios %s\n\n" %(str(mais_dominio[y][1]), str(mais_dominio[y][2]), str(mais_dominio[y][0])))

dados_resumo.write("\n -=-=-=-=-=--=-=--=-=-=-=-=-=-=-=- Faixa dos dominios -=-=-=-=-=--=-=--=-=-=-=-=-=-=-=- \n")
dados_resumo.write("aqui estao todas as faixas presentes dos dominios no alinhamento e qual a proporcao do tamanho da faixa em relacao ao dominio\n\n")
for x in range(len(dominios)):
    dados_resumo.write("\n********************** Dominio %s **********************\n\n" %(str(dominios[x])))
    dados_resumo.write("\ninicio_faixa;;final_faixa;;faixa/maior_dominio;;faixa/menor_dominio\n\n")
    for y in range(len(faixa_posicao_dominio_alinhamento)):
            for z in range(len(faixa_posicao_dominio_alinhamento[y])):
                    ifstr(faixa_posicao_dominio_alinhamento[y][z][0]) == str(dominios[x]) :
                            dados_resumo.write("%s;;%s;;%.2f;;%.2f\n\n" %(str(faixa_posicao_dominio_alinhamento[y][z][1]),
```

```
str(faixa_posicao_dominio_alinhamento[y][z][2]),
float(faixa_posicao_dominio_alinhamento[y][z][5]),
float(faixa_posicao_dominio_alinhamento[y][z][7])))

dados_resumo.close()

#gera matriz com os dominios

cd=(saida + "Dominios_individuais_" + familia)
Dominios_individuais=open(cd, 'w+')
Dominios_individuais.write("#sequencia;;dominio_presente_1;;dominio_presente_2\n")
for x in range(len(individual_dominios)):
    for y in range(len(individual_dominios[x])):
            if y <  (len(individual_dominios[x])-1):
                    Dominios_individuais.write("%s;;" %(str(individual_dominios[x][y])))

            else:
                    Dominios_individuais.write("%s\n" %(str(individual_dominios[x][y])))


Dominios_individuais.close()

#gera matriz com as posicoes nos alinhamentos

cd=(saida + "posicao_alinhamento_" + familia)
posicao_ali=open(cd, 'w+')
posicao_ali.write("#dominio;;ID_sequencia;;posicao_ali_inicio;;posicao_ali_final;;taman
ho_alihamento\n")
for x in range(len(posicao_alinhamento)):
    for y in range(len(posicao_alinhamento[x])):
            if y <  (len(posicao_alinhamento[x])-1):
                    posicao_ali.write("%s;;" %(str(posicao_alinhamento[x][y])))
```

```
            else:
                    aux=int(posicao_alinhamento[x][3])-int(posicao_alinhamento[x][2])
                    posicao_ali.write("%s;;%s\n"          %(str(posicao_alinhamento[x][y]),
str(aux)))
#fala tamanho do dominio na sequencia
posicao_ali.write("#dominio;;ID_sequencia;;tamanho_dominio_sequencia\n")
for x in range(len(tamanho_dominio_sequencia)):
    for y in range(len(tamanho_dominio_sequencia[x])):
            if y <   (len(tamanho_dominio_sequencia[x])-1):
                    posicao_ali.write("%s;;" %(str(tamanho_dominio_sequencia[x][y])))

            else:

                    posicao_ali.write("%s\n" %(str(tamanho_dominio_sequencia[x][y])))
posicao_ali.close()

#gera matriz com as faixas onde os domínios aparecem

cd=(saida + "faixa_posicao_dominio_alinhamento_" + familia)
faixa_dominio_alinhamento=open(cd, 'w+')
faixa_dominio_alinhamento.write("#dominio;;inicio_faixa;;final_faixa\n")
for x in range(len(faixa_posicao_dominio_alinhamento)):
    for y in range(len(faixa_posicao_dominio_alinhamento[x])):
            faixa_dominio_alinhamento.write("%s;;%s;;%s\n"
%(str(faixa_posicao_dominio_alinhamento[x][y][0]),str(faixa_posicao_dominio_alinhame
nto[x][y][1]), str(faixa_posicao_dominio_alinhamento[x][y][2])))

faixa_dominio_alinhamento.close()
```

Apêndice E - Etapas seleção positiva

baixar_cds.sh

```
#!/bin/bash
sequencia=16
for ARQ in $(seq 3 $sequencia)
do
    echo ".....processando AA"
    echo "analisando dados"
    cd /home/pasta_trabalho_"$ARQ"
    echo $(cat arquivo_id.txt | wc -l) sequencias serão baixadas ...
#baixando arquivos
    split -dl190 arquivo_id.txt part
    n_id=0
    echo "baixando do NCBI..."
    total=$(lspart* | wc -l)
    for arquivo in $(lspart* | tr \s \\n)
    do
            n_id=$(echo "$n_id+1" |bc)
            ids=$(cat $arquivo | tr \\n , | cut -d, -f-200)
            list=${ids: :-1}
            wget
"https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=sequences&id=$list&rettype=
fasta_cds_na" -qOfasta.$arquivo
    done
    wait
    cat fasta.* | grep -vP "^$" > cds_seq_sem_repeticao_apenas_fungo_AA"$ARQ".fa
    echo "os arquivos foram salvos em
cds_seq_sem_repeticao_apenas_fungo_AA"$ARQ".fa !"
done
```

comparar_cds_aa.py

```
#!/usr/bin/python3.6
import os
import sys
```

```
importsubprocess

fromBio.SeqimportSeq
fromBioimportSeqIO
fromBio.SeqRecordimportSeqRecord
#formato de entrada script ./script.py
sequencias_aa.fasequencias_cds.facaminho_para_salvar
trabalho = sys.argv
entrada = sys.argv[1]
cds= sys.argv[2]
saida= sys.argv[3]
print
("#################################################################################
################")
print ("entrada : %s" %(str(entrada)))
print ("cds : %s" %(str(cds)))
print ("saida : %s" %(str(saida)))
prot=0
ids = []
print ("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=-=-=-=-=-")
for sequencia_cds in SeqIO.parse(cds, "fasta"):
    if ((len(sequencia_cds.seq)%3) != 0) :
        print("**********essa sequencia tem fase de leitura errada %s"
%(str(sequencia_cds.id)))
print
("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++")
#compara as sequencias_cds com sequencia_aa
for sequencia_cds in SeqIO.parse(cds, "fasta"):
#retira sinal de codon de terminacao e edita o ID para comparar com ID da proteina
    coding_dna = sequencia_cds.seq.translate()
    coding_dna = coding_dna.split("*")[0]
```

```
        aux=0
        prot = prot+1
        id_cds = sequencia_cds.id
        id_cds = id_cds.split("_cds_")[1]
        id_cds = id_cds.split("_1")[0]
    #compara com o banco de dados aa
        for sequencia_aa in SeqIO.parse(entrada, "fasta"):
                ifsequencia_aa.seq == coding_dnaand sequencia_aa.id == id_cds:
                        aux=1
                        simple_seq_r=       SeqRecord(sequencia_cds.seq,       sequencia_aa.id,
    description = "")
                if sequencia_aa.id == id_cds:
                        comparacao=len(coding_dna)/len(sequencia_aa.seq)
                        nome_aa_sequencia = sequencia_aa.id
        ifaux == 1 :
                ids.append(simple_seq_r)
        else:
                print("porcsemelhanca %s cds_sequence %s ERRROOOOOOO aa_sequence
    %s " %(str(comparacao), str(sequencia_cds.id), str(nome_aa_sequencia)))
    #salva o banco de dados limpo
    SeqIO.write(ids, saida, "fasta")
```

entrada_ortho.py

```
    #!/usr/bin/python3.6
    import os
    import sys
    importsubprocess

    fromBio.SeqimportSeq
    fromBioimportSeqIO
    fromBio.SeqRecordimportSeqRecord
    #formato de entrada script  ./script.py sequencia_cdssequencia_aataxo_aacaminho_salvar
    trabalho = sys.argv
    #entrada referencia
```

```
entrada = sys.argv[1]
#arquivo a ser copiado usando a referencia
copia= sys.argv[2]
#taxo a ser copiado usando a referencia
taxo= sys.argv[3]
#nome a ser colocado
saida = sys.argv[4]
#abre os arquivos
cd = ("limpo_cds_ortho_%s.fa" %(str(saida)))
seq_record_1 = open(cd, 'w+')
cd = ("taxonomia_ortho_%s" %(str(saida)))
seq_record_2 = open(cd, 'w+')
seq_record_2.write("Protein_Name;;EC#;;Organism;;GenBank;;Uniprot;;PDB;;Sub_grup
o;;site;;superkingdom;;kingdom;;subkingdom;;phylum;;subphylum;;class;;order;;family;;
genus;;species\n")
#compara os arquivos cds e salva os dados da sequencia editas para entrar no ortho
for sequencia_ent in SeqIO.parse(entrada, "fasta"):
    for sequencia_copia in SeqIO.parse(copia, "fasta"):
            if sequencia_ent.id == sequencia_copia.id :
                    simple_seq_r=  SeqRecord(sequencia_copia.seq,  sequencia_copia.id,
description = "")
                    SeqIO.write(simple_seq_r, seq_record_1, "fasta")
    with open(taxo) as file:
                    for line in file:
                            line=line.rstrip()
                            codigo=line.split(";;")[3]
                            if sequencia_ent.id == str(codigo) :
                                    seq_record_2.write("%s\n" %(str(line)))
seq_record_1.close()
```

filtrar_dominio.py

```
#!/usr/bin/python3.6
import os
import sys
```

```
importre
fromBio.SeqimportSeq
fromBioimportSeqIO
fromBioimportAlignIO
trabalho = sys.argv
#arquivo de selecao positiva em fasta
entrada = sys.argv[1]
#dominios considerados
dominios = sys.argv[2]
#sequencia aa
seq_aa = sys.argv[3]
#sequencia cds
seq_cds = sys.argv[4]
#taxonomia
taxo = sys.argv[5]
#arquivo para salvar
saida = sys.argv[6]
variacao_tamanho_dominio=0.5
tabela=[]
#carrega a tabela no computador
with open(entrada) as file:
        for line in file:
                line=line.rstrip()
                linha=line.split(";;")
                tabela.append(linha)
#carrega os dominios considerados relevantes que se quer analisar
tabela_dominio=[]
with open(dominios) as file:
        for line in file:
                line=line.rstrip()
                tabela_dominio.append(line)
#pega o tamanho dos dominios
tabela_tam_dominios = []
```

```
for i in range(len(tabela_dominio)):
    for seq_record in SeqIO.parse(seq_aa, "fasta"):
            controle=0
            aux=[]
            menor_dominio=1000000
            maior_dominio=0
            for j in range(len(tabela)):
                    ifstr(tabela[j][0])  ==  str(seq_record.id)  andstr(tabela[j][6])  ==
str(tabela_dominio[i]) :
                            controle=1
                            aux_id = str(seq_record.id)
                            aux_dominio = str(tabela_dominio[i])

                            ifmenor_dominio>int(tabela[j][1]):
                                    menor_dominio = int(tabela[j][1])
                            ifmaior_dominio<int(tabela[j][2]):
                                    maior_dominio = int(tabela[j][2])
            if controle == 1:
                    tamanho_dominio = maior_dominio - menor_dominio+1
                    aux.append(aux_id)
                    aux.append(aux_dominio)
                    aux.append(tamanho_dominio)
            ifaux != [] :
                    tabela_tam_dominios.append(aux)
#tira a media de cada dominio
media_dominios = []
for i in range(len(tabela_dominio)):
    soma=0
    contador=0
    media=0
    aux=[]
    for j in range(len(tabela_tam_dominios)):
            iftabela_tam_dominios[j][1] == tabela_dominio[i]:
```

```
                    soma=soma+tabela_tam_dominios[j][2]
                    contador=contador+1
    media=int(soma/contador)
    aux.append(tabela_dominio[i])
    aux.append(media)
    media_dominios.append(aux)
#analisa as sequencias com os dominios e se eles estao dentro da media
seq_selecionada=[]
for seq_record in SeqIO.parse(seq_aa, "fasta"):
    aux=[]
    aux.append(seq_record.id)
    controle=1
#ve as sequencias com dominios corretos
    for i in range(len(media_dominios)):
        tamanho_max_dominio = media_dominios[i][1] + media_dominios[i][1]*variacao_tamanho_dominio
        tamanho_min_dominio = media_dominios[i][1] - media_dominios[i][1]* variacao_tamanho_dominio
        for j in range(len(tabela_tam_dominios)):
            ifstr(tabela_tam_dominios[j][0]) == str(seq_record.id) andstr(tabela_tam_dominios[j][1]) == media_dominios[i][0]:
                ifint(tabela_tam_dominios[j][2]) <int(tamanho_min_dominio) orint(tabela_tam_dominios[j][2]) >int(tamanho_max_dominio):
                    fora_padrao=0
                else:
                    aux.append(media_dominios[i][0])
#pega as sequencias com dominios corretos
    for i in range(len(media_dominios)):
        ifmedia_dominios[i][0] in auxand controle == 1:
            controle_correto=1
        else:
            controle=0
    if controle == 1:
```

```
            seq_selecionada.append(aux[0])
##############################################################################
##########
nome_seq_aa = seq_aa.split("/")
cd=(saida + "seq_sem_dados_esp_" + nome_seq_aa[len(nome_seq_aa)-1])
dados_seq=open(cd, 'w+')
#gera os arquivos de entrada sem os dados espurios
for seq_record in SeqIO.parse(seq_aa, "fasta"):
    for i in range(len(seq_selecionada)):
            ifstr(seq_selecionada[i]) == str(seq_record.id):
                    SeqIO.write(seq_record, dados_seq, "fasta")
dados_seq.close()
nome_seq_cds = seq_cds.split("/")
cd=(saida + "cds_sem_dados_esp_" + nome_seq_cds[len(nome_seq_cds)-1])
dados_cds=open(cd, 'w+')
#gera os arquivos de entrada sem os dados espurios
for seq_record in SeqIO.parse(seq_cds, "fasta"):
    for i in range(len(seq_selecionada)):
            ifstr(seq_selecionada[i]) == str(seq_record.id):
                    SeqIO.write(seq_record, dados_cds, "fasta")
dados_cds.close()
nome_taxo = taxo.split("/")
cd=(saida + "taxo_sem_dados_esp_" + nome_taxo[len(nome_taxo)-1])
dados_taxo=open(cd, 'w+')
dados_taxo.write("Protein_Name;;EC#;;Organism;;GenBank;;Uniprot;;PDB;;Sub_grupo;;
site;;superkingdom;;kingdom;;subkingdom;;phylum;;subphylum;;class;;order;;family;;gen
us;;species\n")
#gera os arquivos de entrada sem os dados espurios
with open(taxo) as file:
    for line in file:
            line=line.rstrip()
            linha=line.split(";;")
            for i in range(len(seq_selecionada)):
```

```
                    ifstr(seq_selecionada[i]) in str(linha):
                        dados_taxo.write("%s\n" %(str(line)))
dados_taxo.close()
```

agrupar_por_especie.py

```
#!/usr/bin/python3.6
import os
import sys
importsubprocess
fromBio.SeqimportSeq
fromBioimportSeqIO
fromBio.SeqRecordimportSeqRecord
#formato de entrada script  ./ortho.py sequencias/AA+_eukaryota_sequencia.fa taxo/ nome_que_se_quer_salvar
trabalho = sys.argv
entrada = sys.argv[1]
taxo= sys.argv[2]
saida= sys.argv[3]
#cria matriz com as especies existentes no arquivo
print("\n########################\n")
dados1=open(taxo).readlines()
x1=[]
for i in range(len(dados1)):
    if i ==0:
            linha=dados1[i].rstrip()
            if "NOME" in linha:
                    titulo=linha.split(';;')[17+1]
            else:
                    titulo=linha.split(';;')[17]
    if i !=0:
            linha=dados1[i].rstrip()
            linha=linha.split(';;')[17]
            aux=0
            for j in range(len(x1)):
```

```
                    if linha in x1[j]:
                        aux=1
                        x1[j][1]=int(x1[j][1]) +1
            ifaux == 0:
                    x1.append([])
                    (x1[len(x1)-1]).append("%s" %str(linha))
                    (x1[len(x1)-1]).append(1)
#Cria matriz dos generos que tem especies
print("\n####################### 222222\n")
dados1=open(taxo).readlines()
x2=[]
for i in range(len(dados1)):
    if i ==0:
            linha=dados1[i].rstrip()
            if "NOME" in linha:
                    titulo=linha.split(';;')[17+1]
            else:
                    titulo=linha.split(';;')[17]
    if i !=0:
            linha=dados1[i].rstrip()
            linha_especie=linha.split(';;')[17]
            linha_genero=linha.split(';;')[16]
            aux=0
            for j in range(len(x2)):
                    iflinha_genero in x2[j] andlinha_especie != "vazio":
                        aux=1
            ifaux == 0 andlinha_especie != "vazio":

                    (x2).append("%s" %str(linha_genero))
#Cria matriz dos generos que com vazio e que nao tem representante com especie
print("\n####################### 33333333\n")
dados1=open(taxo).readlines()
x3=[]
```

```
for i in range(len(dados1)):

    if i ==0:
            linha=dados1[i].rstrip()
            if "NOME" in linha:
                    titulo=linha.split(';;')[17+1]
            else:
                    titulo=linha.split(';;')[17]

    if i !=0:
            linha=dados1[i].rstrip()
            linha_especie=linha.split(';;')[17]
            linha_genero=linha.split(';;')[16]
            aux=0
            for j in range(len(x3)):
                    iflinha_genero in x3[j] andlinha_especie == "vazio":
                            aux=1

            ifaux == 0 andlinha_especie == "vazio" andlinha_genero != "vazio":

                    iflinha_genero in x2:
                            controle =0
                    else:
                            (x3).append("%s" %str(linha_genero))
#############################################################################
#####

#cria arquivo para cada espécie ignorando os vazios
print(x1)
for i in range(len(x1)):
    ids = []
    for sequencia in SeqIO.parse(entrada, "fasta"):
            with open(taxo) as file:
```

```
                for line in file:
                        line=line.rstrip()
                        linha=line.split(";;")[3]
                        specie=line.split(";;")[17]
                        ifstr(sequencia.id) == str(linha) andstr(specie) == str(x1[i][0])
andstr(x1[i][0]) != "vazio":
                                simple_seq_r= SeqRecord(sequencia.seq, sequencia.id,
description = "")
                                ids.append(simple_seq_r)
    ifstr(x1[i][0]) != "vazio":
            nome_arq=x1[i][0]
            nome_arq=nome_arq.replace(" ","_")
            local_savar=saida+nome_arq +".fa"
            SeqIO.write(ids, local_savar, "fasta")
#cria arquivo para genero que nao tem especie
print(x3)
for i in range(len(x3)):
    ids = []
    for sequencia in SeqIO.parse(entrada, "fasta"):
            with open(taxo) as file:
                    for line in file:
                            line=line.rstrip()
                            linha=line.split(";;")[3]
                            specie=line.split(";;")[17]
                            genus=line.split(";;")[16]
                            ifstr(sequencia.id) == str(linha) andstr(specie) == "vazio"
andstr(genus) == str(x3[i]):
                                simple_seq_r= SeqRecord(sequencia.seq, sequencia.id,
description = "")
                                ids.append(simple_seq_r)
    ifstr(x3) != "vazio":
            nome_arq="x"+x3[i]
            local_savar=saida+nome_arq +".fa"
```

```
            SeqIO.write(ids, local_savar, "fasta")
```

renomear.sh

```
#!/bin/bash
FAM=${1}
PAM=${2}
arquivo=$(ls $FAM -1)
#echo "$arquivo"
contador="0"
echo "" >abreviacao
for ARQ in $arquivo
do
    contador=$(echo "$contador + 1" | bc )
    abre=$(echo "${ARQ:0:2}$contador")
    #echo "$FAM$ARQ"
    mv "$FAM$ARQ" "$FAM$abre.fa"
    mv "$PAM$ARQ" "$PAM$abre.fa"
    echo "$abre.fa ---> $ARQ" >>abreviacao
done
```

conf_potion.conf

```
###############PROJECT PARAMETERS#########################

mode = site                    # mainanalysismode. Currently POTION supportsonly site-
modelsanalysis.

CDS_dir_path = /home/cds_sequences/

homology_file_path = /home/new_groups_OrthoMCL.txt

project_dir_path = /home/results_AA/

max_processors = 4
```

```
remove_identical = yes                                  # "yes" to remove 100% identicalnucleotidegroupsattheverybeginningof
                        # analysis, "no" otherwise

verbose = 1                     # 1 to print nice log messagestellingyouwhatisgoingon. 0 otherwise

############SEQUENCE/GROUP PARAMETERS################

groups_to_process = all             # Defines whichlinesofthe cluster file (orthologgroups) willbeprocessed.
                        # Use "all" toprocesseverygroup, "-" to set groupsbetweentwogivenlines
                        # (includingthesaidlines).
                        # Use "!" tonotprocess a specificline, canbeusedwith "-" tospecify a
                        # set tonotbeprocessed. Usefulifgroups are taking too longtofinish.
                            # Use "," or ";" to set distinct sets
                            # Examples: 1;4-10;12  willprocessgroups 1, 4 to 10 andgroup 12
                            #       all;!3    willprocessallgroups, exceptgroup 3
                            #       all;!3-5  willprocessallgroups, exceptgroups 3 to 5

behavior_about_bad_clusters = 1     # whatshould POTION do if it finds a cluster with a sequenceremoved
                        # duetoanyfilter? Possibleoptions are:
                        # 0 - does notfilteranysequence (notrecommended)
                        # 1 - removalofanyflaggedsequence
                        # 2 - removalofanygroupwithflaggedsequences
```

```
behavior_about_paralogs = 1          # thisvariablecontrols for what POTION will do if a groupwithparalogous
                                     # genes isfound. Possibleoptions are:
                                     # 0 - analyzeallsequenceswithingroup
                                     # 1 - remove allparalogouswithingroup, analyzingonly single-copy genes
                                     # 2 - remove groupswithparalogous genes
                                     # 3 - remove single-copy genes, analyzingallparalogouswithingrouptogether
                                     # 4 - remove single-copy genes and split remainingparalogousinto individual
                                     # species, evaluatingeachsubgroupindividually

validation_criteria = all            # qualitycriteriato remove sequences. Possiblevalues are:
                                     # 1 - checks for valid start codons
                                     # 2 - checks for valid stop codons
                                     # 3 - checks for sequencesizemultipleof 3
                                     # 4 - checks for nucleotidesoutside ATCG
                                     # 'all' applieseveryverification

additional_start_codons = ()         # thesecodons, plus theonesspecified in codontable, willbethevalid start
                                     # codons for validationpurposes
additional_stop_codons = ()          # same as start codons

codon_table = 1

absolute_min_sequence_size = 150     # minimumsequencelengthcutoff for sequence/groupfurtherevaluation

absolute_max_sequence_size = 10000   # maximumsequencelengthcutoff for sequence/groupfurtherevaluation
```

relative_min_sequence_size = 0.83 # sequencessmallerthanmean|meadian times thisvaluewillbefiltered

relative_max_sequence_size = 1.2 # sequencesgreaterthanmean|meadian times thisvaluewillbefiltered

sequence_size_average_metric = mean # whichaveragemetricwillbecalculatedto determine the
# minimum/maximumrelativelengths ranges for sequenceremoval
# Possiblevalues are "mean" and "median"

min_group_identity = 30 # meanminimumgroupidentitycutoff in pairwisesequencealignments

max_group_identity = 100 # meanmaximumgroupidentitycutoff in pairwisesequencealignments

group_identity_comparison = aa # thekindofsequencethatwillbeusedwhencomputingmeangroupidentity
# possiblevalues are "nt" or "aa"

min_sequence_identity = 30 # minimum (mean/median) sequenceidentitycutoff in pairwisesequencealignments

max_sequence_identity = 100 # maximum (mean/median) sequenceidentitycutoff in pairwisesequencealignemnts

sequence_identity_average_metric = mean # wouldyou like to use meanormediantomeasuresequenceidentity?
# possiblevalues are "mean" and "median"

sequence_identity_comparison = aa # thekindofsequencethatwillbeusedwhencomputingsequenceidentity
# possiblevalues are "nt" and "aa"

min_gene_number_per_cluster = 3 # minimum # genes in groupafterallfilteringsteps

max_gene_number_per_cluster = 50 # maximum # genes in groupafterallfilteringsteps

min_specie_number_per_cluster = 3 # minimum # species in groupafterallfilteringsteps

max_specie_number_per_cluster = 50 # maximum # species in groupafterallfilteringsteps

reference_genome_file = # genomereferencename, leaveblank for none (samenameused in fasta file)

homology_filter = 1
############THIRD-PARTY SOFTWARE CONFIGURATIONFIRST#################

multiple_alignment = prank # programused for multiplesequencealignment. Possiblevalues are
# muscle, mafftandprank

bootstrap = 100 # numberofbootstraps in phylogeneticanalysis

phylogenetic_tree_speed = fast # fast orslowanalysis? Used in phylipdnamlorpromlonly

phylogenetic_tree = proml # programused for phylogenetictreereconstruction. Possiblevalues are
# promldnaml, phyml_aaandphyml_nt

recombination_qvalue = 0.1 # q-value for recombinationdetection. Must occur for allthespecifiedtests,
# or 0 | N.A. toskiprecombinationtest

rec_minimum_confirmations = 2 # minimumnumberofsignificantrecombinationtests positives (1-3), or N.A. to
# skiprecombinationtest

rec_mandatory_tests = phi # anycombinationofthethreetestnames, separatedbyspaces, or N.A. to use
# anytest

remove_gaps = strict # numericvaluesbetween 0 and 1 will remove columnswiththatpercentageof
# gaps. Valuesof "strict" or "strictplus" will use respectivelythese
# filtersto remove unreliableregions (described in trimalarticle)

PAML_models = m12 m78 # codemlmodelstobegenerated. "m12" and/or "m78" valuesacceptable.
pvalue = 0.05 # p-values for positive selectiondetection
qvalue = 0.05 # q-values for positive selectiondetection

dados_grupo_selecao_positiva.py

```
#!/usr/bin/python3.6
import os
import sys
importre
fromBio.SeqimportSeq
fromBioimportSeqIO
fromBioimportAlignIO
trabalho = sys.argv
#arquivo de selecao positiva em fasta
```

```
entrada = sys.argv[1]
#tabela de saida do pfam editado para entrada de programa
tabela_editada_pfam=sys.argv[2]
#sequencias da familia
sequencia=sys.argv[3]
#posicao_alinhamento - aquela saida do analise de dominiospfam
posicao_alinhamento=sys.argv[4]
#local para salvar
saida = sys.argv[5]
#nome da familia no inicio do arquivo
familia = sys.argv[6]

tabela_dominio=[]
dados1=open(entrada).readlines()
contador=0
linha_limpa=[]
out_potion_editado=[]
#pega a saida do potion e edita de forma mais facil de trabalhar
for j in range(len(dados1)):
        contador=contador+1
        line=dados1[j].rstrip()

        if contador == 1:
                sequencias_selecao_positiva=line.split(": ")[1]
                sequencias_selecao_positiva=sequencias_selecao_positiva.split(" ")
        if contador == 2:
                model=line.split("(")[1]
                model=model.split(")")[0]
                model=model.replace(" ","_")
                line=line.split(" ")[0]

                line=line.split(">")[1]
                linha_limpa.append(line)
```

```
        if contador == 5:
                linha_limpa.append(model)
                linha_limpa.append(sequencias_selecao_positiva)
                contador=0
                out_potion_editado.append(linha_limpa)
                linha_limpa=[]
#carrega tabela editada do pfam
tabela=[]
with open(tabela_editada_pfam) as file:
                for line in file:
                        line=line.rstrip()
                        linha=line.split(";;")
                        iffloat(linha[12]) <= float(9e-4):
                                tabela.append(linha)
#carrega posicao alinhamento
posi_ali=[]
with open(posicao_alinhamento) as file:
        cont=0
        for line in file:
                if "#" in line:
                        cont=cont+1
                if "#" in lineorcont == 2:
                        aux=0
                else:
                        line=line.rstrip()
                        linha=line.split(";;")
                        posi_ali.append(linha)
#salva dados na sequencia
dados_tabela_1=[]
for i in range(len(out_potion_editado)):
        for j in range(len(out_potion_editado[i][2])):
                aux=[]
```

```
                aux.append(out_potion_editado[i][0])
                aux.append(out_potion_editado[i][1])
                for seq_record in SeqIO.parse(sequencia, "fasta"):
                        ifstr(out_potion_editado[i][2][j]) in str(seq_record.id):
                                seq_ref=seq_record.seq
                                seq_ref=seq_ref.upper()
                                aux.append(seq_record.id)
                                aux.append(len(seq_ref))
                for k in range(len(tabela)):
                        ifstr(out_potion_editado[i][2][j]) in str(tabela[k][0]):
                                aux.append(tabela[k][6])
                                aux.append(tabela[k][1])
                                aux.append(tabela[k][2])
                aux.append(out_potion_editado[i][2][j])
                dados_tabela_1.append(aux)
for x in range (len(dados_tabela_1)):
        print(dados_tabela_1[x])
#agora compara os alinhamentos
dados_tabela_2=[]
for i in range(len(out_potion_editado)):
        for j in range(len(out_potion_editado[i][2])):
                aux=[]
                aux.append(out_potion_editado[i][0])
                aux.append(out_potion_editado[i][1])
                for k in range(len(posi_ali)):
                        ifstr(out_potion_editado[i][2][j]) in str(posi_ali[k][1]):
                                aux.append(posi_ali[k][0])
                                aux.append(posi_ali[k][2])
                                aux.append(posi_ali[k][3])
                aux.append(out_potion_editado[i][2][j])
                dados_tabela_2.append(aux)
#agora acha os dominios presentes
dominios=[]
```

```
for i in range(len(out_potion_editado)):
	aux=[]
	aux_vetor=[]
	aux.append(str(out_potion_editado[i][0]))
	aux.append(str(out_potion_editado[i][1]))
	for x in range(len(tabela)):
		aux_1=0
		seq=tabela[x][0].split(".")[0]
		ifstr(seq) in str(out_potion_editado[i][2]):
			ifstr(tabela[x][6]) in str(aux_vetor):
				aux_1=0
			else:
				aux_vetor.append(str(tabela[x][6]))
	aux.append(aux_vetor)
	dominios.append(aux)
cd=(saida + "dados_selecao_positiva_" + familia)
selecao_positiva_dominio=open(cd, 'w+')

for i in range(len(out_potion_editado)):
	selecao_positiva_dominio.write("#sequencia_model;;model;;tamanho_seq;;dominio;;s
tart_seq;;end_seq;;tamanho_dominio;;\n")
	selecao_positiva_dominio.write("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=\n")
	selecao_positiva_dominio.write("----%s    %s----\n"    %(str(out_potion_editado[i][0]),
str(out_potion_editado[i][1])))
	selecao_positiva_dominio.write("*******************dominios    presentes    na
selecao positiva\n")

	for j in range(len(dominios)):
		ifout_potion_editado[i][0] == dominios[j][0] andout_potion_editado[i][1] ==
dominios[j][1]:
			dominios_dentro_selecao= dominios[j][2]
			for k in range(len(dominios[j][2])):
```

```
                                selecao_positiva_dominio.write("%s\n"
%(str(dominios[j][2][k])))
#sequencia
        selecao_positiva_dominio.write("********************dados sequencia\n")
        for j in range(len(dados_tabela_1)):
                ifout_potion_editado[i][0]          ==          dados_tabela_1[j][0]
andout_potion_editado[i][1] == dados_tabela_1[j][1]:
                        selecao_positiva_dominio.write("%s;;"
%(str(dados_tabela_1[j][len(dados_tabela_1[j])-1]))
                        selecao_positiva_dominio.write("%s;;" %(str(dados_tabela_1[j][1])))
                        selecao_positiva_dominio.write("%s;;" %(str(dados_tabela_1[j][3])))
                        for k in range(len(dominios_dentro_selecao)):
                                menor_dominio=1000000
                                maior_dominio=0
                                for l in range(len(dados_tabela_1[j])):
                                        ifstr(dados_tabela_1[j][l])                    ==
str(dominios_dentro_selecao[k]):
                                                ifmenor_dominio>int(dados_tabela_1[j][l+1]):
                                                        menor_dominio                  =
int(dados_tabela_1[j][l+1])
                                                ifmaior_dominio<int(dados_tabela_1[j][l+2]):
                                                        maior_dominio                  =
int(dados_tabela_1[j][l+2])
                                ifstr(dominios_dentro_selecao[k]) in str(dados_tabela_1[j]):
                                        selecao_positiva_dominio.write("%s;;"
%(str(dominios_dentro_selecao[k])))
                                        selecao_positiva_dominio.write("%s;;"
%(str(menor_dominio)))
                                        selecao_positiva_dominio.write("%s;;"
%(str(maior_dominio)))
                                        tamanho= maior_dominio-menor_dominio+1
                                        selecao_positiva_dominio.write("%s;;"
%(str(tamanho)))
```

```
                    selecao_positiva_dominio.write("\n")

#alinhamento
        selecao_positiva_dominio.write("********************dados alinhamento\n")
        for j in range(len(dados_tabela_2)):
                ifout_potion_editado[i][0]          ==          dados_tabela_2[j][0]
andout_potion_editado[i][1] == dados_tabela_2[j][1]:
                    selecao_positiva_dominio.write("%s;;"
%(str(dados_tabela_2[j][len(dados_tabela_2[j])-1]))
                    selecao_positiva_dominio.write("%s;;" %(str(dados_tabela_2[j][1])))

                    for k in range(len(dominios_dentro_selecao)):
                        menor_dominio=1000000
                        maior_dominio=0
                        for l in range(len(dados_tabela_2[j])):
                            ifstr(dados_tabela_2[j][l])                    ==
str(dominios_dentro_selecao[k]):
                                ifmenor_dominio>int(dados_tabela_2[j][l+1]):
                                    menor_dominio                    =
int(dados_tabela_2[j][l+1])
                                ifmaior_dominio<int(dados_tabela_2[j][l+2]):
                                    maior_dominio                    =
int(dados_tabela_2[j][l+2])
                        ifstr(dominios_dentro_selecao[k]) in str(dados_tabela_2[j]):
                            selecao_positiva_dominio.write("%s;;"
%(str(dominios_dentro_selecao[k])))
                            selecao_positiva_dominio.write("%s;;"
%(str(menor_dominio)))
                            selecao_positiva_dominio.write("%s;;"
%(str(maior_dominio)))
                            tamanho= maior_dominio-menor_dominio+1
                            selecao_positiva_dominio.write("%s;;"
%(str(tamanho)))
```

```
                    selecao_positiva_dominio.write("\n")
        selecao_positiva_dominio.write("**********visao geral: sem dominio\n")
        for j in range(len(dominios_dentro_selecao)):
            selecao_positiva_dominio.write("###############%s\n\n" %(str(dominios_dentro_selecao[j])))
            for k in range(len(dados_tabela_1)):
                ifout_potion_editado[i][0] == dados_tabela_1[k][0] andout_potion_editado[i][1] == dados_tabela_1[k][1]:
                    ifstr(dominios_dentro_selecao[j]) in str(dados_tabela_1[k]):
                        a=0
                    else:
                        selecao_positiva_dominio.write("a sequencia %s não tem o dominio %s\n" %(str(dados_tabela_1[k][len(dados_tabela_1[k])-1]), str(dominios_dentro_selecao[j])))
selecao_positiva_dominio.write("\n\n")
selecao_positiva_dominio.close()
```

Apêndice F - Revisão da literatura para descrever os domínios

1- O domínio CDH-cyt consiste de um domínio citocromo b, citado na literatura tendo em média 256 aminoácidos (ZÁMOCKÝ *et al.*, 2004). Esse domínio é encontrado associado a estrutura da enzima Cellobiosedehydrogenases, porém apresenta atividade catalítica mesmo separado enzimaticamente do complexo. O CDH-cyt pode ser encontrado associado a outros domínios como CBM apenas ou com outros tipos de domínios desconhecidos (HALLBERG *et al.*, 2000; LEVASSEUR *et al.*, 2013).

2- O domínio GMC_oxred_N consiste de um domínio de ligação ao FAD que corresponderia ao descrito do domínio na estrutura 1COX, sendo formado por três segmentos de sequências não-contínuos e intercalados por dois segmentos não-contínuos que correspondem ao domínio de interação ao substrato. A região de interação entre esses dois domínios formam a região catalítica da enzima. O domínio GMC_oxred_C corresponde à região de ligação ao substrato (VRIELINK *et al.*, 1991.

3- Na biologia molecular, um módulo de ligação a carboidratos (CBM) é um domínio proteico encontrado em enzimas ativas a carboidratos (por exemplo, glicosil hidrolases). A família de módulos de ligação a carboidratos 1 (CBM1) apresenta 36 aminoácidos. Esse domínio contém 4 resíduos de cisteína conservados que estão envolvidos na formação de duas ligações dissulfeto (PROSITE, 2019). A degradação microbiana de celulose e xilanos requer vários tipos de enzimas, como endoglucanases (EC 3.2.1.4), celobiohidrolases (EC 3.2.1.91), exoglucanases ou xilanases (EC 3.2.1.8). Celulases e xilanases geralmente consistem em um domínio catalítico e um domínio de ligação à celulose (CBD), também chamado de módulo de ligação a carboidratos (CBM) (PROSITE, 2019).

4- O domínio DsbA é uma dissulfeto oxidoredutase de tiol, o qual é parte de uma família de proteínas oxidoredutase dissulfeto que compartilham uma estrutura de domínio comum conhecida como dobra da tioredoxina. A dobra da tioredoxina é encontrada em cinco classes distintas de proteínas, com propriedades parecidas e que interagem com substratos contendo cisteína (GUDDAT *et al.*, 1998). Ela consiste de uma folha beta de quatro filamentos e três hélices alfas (MARTIN, 1995).

5- O domínio Pkinase consiste em um domínio da proteína cinase estruturalmente conservado em eucariotos, tendo função catalítica consistindo em um pequeno subdomínio N-terminal, formado principalmente por folhas beta, e um sub domínio C-terminal maior formado principalmente por alfa-hélices. A região de ligação do ATP fica entre esses subdomínios mudando de conformação dependendo da presença do ATP (SCHEEFF *et al.*, 2005). As cinases apresentam uma rica diversidade estrutural, modos de ligação e especificidades de substrato, mas apresentam características estruturais em comum. O motivo estrutural conservado fornece indicações que essa enzima transfere o fosfato de um trifosfato de nucleotídeo de purina para um grupo hidroxila do substrato (HANKS *et al.*, 1995).

6- O domínio RWD é um domínio presente na GCN2. A GCN2 é uma subunidade alfa do fator de iniciação de tradução (eIF2alfa). GCN2 requer a interação do GCN2 com GCN1, que ocorre na região terminal N do GCN2 (NAMEKI *et al.*, 2004). O domínio RWD pode estar presente em outras proteínas, apresentando funções não conhecidas e diversas. Um exemplo é a proteína RWDD3, que apresenta o domínio RWD que se liga com a proteína Ubc9 (NAMEKI *et al.*, 2004).

7- O domínio HET é uma região conservada de aproximadamente 150 aminoácidos que está presente em proteínas de incompatibilidade heterocariótica. Essa incompatibilidade é muito comum em fungos filamentosos, impedindo a fusão dos hifas de isolados diferentes. A especificidade das diferentes proteínas HET parecem estar envolvidas com repetições do domínio WD40 (ESPAGNE *et al.*, 2002).

8- O domínio PhyH consiste em uma família de proteínas composta pela proteína eucariótica fitoil-CoA-dioxygenase, ectoína-hydroxylases e várias desoxigenases bacterianas. Essas enzimas atuam na hidroxilação de fitanoil-CoA, envolvida na etapa de decomposição de ácido fitânico e ectoine, uma molécula de interesse biotecnológico (JANSEN, 2000; PRABHU *et al.*, 2004).

9- As Aspartil proteases (Asp) são enzimas proteolíticas amplamente distribuídas, sendo que os fungos contêm uma variedade elevada de sequências, com algumas bem caracterizadas e com utilização na indústria, porém pouco se sabe sobre seu papel fisiológico. Em fungos, as

Asppodem ser encontradas em vacúolos e associadas a membrana celular através de glicosilfosfatidilinositol (REVUELTA *et al.*, 2014).

10- A FAD_binding_3 representa um domínio de ligação ao FAD de monoxigenases. Essas enzimas podem usar além do FAD, metais de transição para gerar radicais hidroxilas e inseri-la nas cadeias carbônicas. Essas enzimas estão presentes em rotas metabólicas como de degradação de anéis aromáticos, estando presente em uma gama grande de seres vivos (HARAYAMA *et al.*, 1992).

11- O domínio FAD_binding_4 compreende um domínio de ligação ao FAD. Na estrutura da 1VAO, o domínio de interação com o FAD se encontra no intervalo 6 - 270 e 500 - 560 (MATTEVI *et al.*, 1997). Analisando-se os dados no Uniprot P56216, referente à estrutura 1VAO, se percebe que a região considerada pelo Pfam como sendo o domínio FAD_binding_4 compreende os aminoácidos 71 - 213, a qual abrange a região com vários aminoácidos que interagem com o FAD.
O domínio FAD-oxidase_C compreende a segunda região do domínio de interação a FAD, contendo, em seu intervalo, a histidina de ligação covalente ao FAD. A estrutura 1VAO apresentou a região do domínio no intervalo 314 - 546. Essa região realiza muitas interações com o substrato, contendo também os aminoácidos prováveis de catalisar a reação (MATTEVI *et al.*, 1997).

12- A Glyoxal_oxid_N corresponde a uma região característica da família Glyoxal oxidase, presente no N-terminal e que compreende cerca de 300 resíduos (PFAM, 2019). Modelos de estrutura da Glyoxal oxidase mostram certas semelhanças de conservação estrutural e de aminoácidos conservados com a galactose oxidase (FIRBANK *et al.*, 2001) e a galactose oxidase é semelhante a álcool oxidase, estando juntas formando a AA5 subfamília 2, que apresenta uma região conservada entre os aminoácidos 1 ao 370, exibindo sete domínios Kelch. Poucos dados estruturais foram encontrados para a Glioxal oxidase, mas é possível correlacionar seus domínios com enzimas próximas, usando dados como a posição de aminoácidos catalíticos com posições próximas. Dessa forma, a descrição da região da família Glioxal oxidase assemelha-se muito com a região da Alcóol oxidases (DAOU *et al.*, 2017; YIN *et al.* 2015)..

13- O domínio DUF1929 é considerado uma estrutura de domínio, referenciada pelo Pfam no trabalho de Firbank*et al.* (2001), identificando essa estrutura como sendo o domínio III, cuja função não é entendida, mas o domínio é conservado. Esse domínio consiste de conjunto de sete fitas β, antiparalelas, em torno de um núcleo hidrofóbico (FIRBANK *et al.*, 2001). Analisando-se os dados do acesso Uniprot E3QHV8, referentes a estrutura 5C92, notou-se que a região do domínio DUF1929 abrange os aminoácidos 408 - 504, sendo a mesma região descrita do domínio C-terminal que contém a histidina de coordenação do cobre (YIN *et al.*, 2015).

14- A Chitin_bind_1 e CBM32 consistem de domínios CBM. A maioria desses domínios apresenta atividade de ligação apolissacarídeos, existindo dezenas de CBMs diferentes que se ligam a substratos diferentes. Esse domínio encontra-se em muitas enzimas que tem ação em carboidratos, podendo ser hidrolíticas ou não, como celulases, xilanases e celobiose desidrogenases. O tamanho dos domínios CBM pode variar de 30 até cerca 200 aminoácidos e podem apresentar estruturas tridimensionais próximas. Sua capacidade de interação ao substrato pode ser atribuída aos aminoácidos aromáticos que formam uma superfície hidrofóbica (SHOSEYOV *et al.*, 2006). A remoção da CBM pode levar a redução ou perda de função de diversas GH (glicosil hidrolases). Essa redução é principalmente no substrato insolúvel, visto que a redução em substrato solúvel normalmente não é afetada. O CBM pode ser encontrado em praticamente todo os organismos, estando associado com funções de reconhecimento de polissacarídeos (GUILLÉN *et al.*, 2010).

15- O domínio WSC é presente em várias proteínas diferentes, apresentando distintas funções. Foi identificado inicialmente em *Saccharomyces cerevisiae*, apresentando o papel de sensor de estresse, como alta temperatura, e em *Pichia pastoris*, tendo a função de detectar metanol. No caso do domínio WSC em álcool oxidase, demonstrou-se que tem a função de interação com o substrato. Nos testes realizados por Oide*et al.* (2019) foi mostrado a interação mais elevada com xilano de madeira, mas especulado que poderiam existir domínios WSC para diversos tipos de polissacarídeos como ocorre com as CBM (GUILLÉN *et al.*, 2010).

16- Podoplanin consiste de uma região comum em proteínas semelhantes a podoplanina presentes em mamíferos. Essa proteína controla a forma dos podócitos, controlando a seletividade da excreção nos rins (BOUCHEROT, 2002).

17- O domínio F5_F8_type_C é encontrado em proteínas sanguíneas de mamíferos, estando associado com domínios de interação com a membrana fosfolipídica da superfície plaquetária (VEERARAGHAVAN *et al.*, 1998). Esse domínio apresenta cerca de 150 aminoácidos, se assemelhando com outros domínios de ligação a fosfolipídeos.

18- O domínio Kelch está presente em várias proteínas em diferentes grupos de organismos, amplamente encontrada, formada por aproximadamente 50 aminoácidos, estando presente com cinco à sete cópias, que se curvam e formam uma estrutura solenóide, existindo várias isoformas já identificadas do domínio kelch (ADAMS *et al.*, 2000). O domínio Kelch está presente nas estruturas da galactose oxidase e da álcool oxidase. Na álcool oxidase ocorrem sete cópias do domínio kelch, sendo um domínio de interação ao cobre e de atividade catalítica (ITO *et al.*, 1994; YIN *et al.*, 2015).

19- O domínio PAN_1 é presente em pré-calicreína e plasminogênio, que são serina proteases presente em mamíferos. A plasminogen tem função na dissolução de coágulos de fibrina (LAW *et al.*, 2013; TORDAI *et al.*, 1999). Esse domínio está presente em outras proteínas, existindo em vários organismos como bactérias, fungos, plantas e animais, intermediando interações proteína-proteína ou proteína-carbohidrato (GONG *et al.*, 2012).

20- O domínio flavodoxin-like é uma estrutura de cerca de 170 aminoácidos, envolvida na ligação ao mononucleotídeo de flavina (FMN), sendo os menores membros da família de flavoproteínas e estando envolvida em reações de transferência de elétrons (DRENNAN *et al.*, 1999; WAKABAYASHI *et al.*, 1989). A flavodoxin_1 está presente em organismos procarióticos, podendo ser encontrado em eucariotos como algas, ou estruturas similares em outros eucariotos (DRENNAN *et al.*, 1999; ZHAO *et al.*, 2008).

21- O domínio FMN_red, realiza a interação com FMN de forma não-covalente e tem uma dobra estrutural parecida com as flavodoxinas. Esse domínio está envolvido com a redução de NADPH (DELLER *et al.*, 2006) e está correlacionado a enzimas em bactérias, fungos e plantas. As enzimas que apresentam esse domínio apresentam atividades diversas, como redução de quinonas e de cromo VI (ACKERLEY *et al.*, 2004; GRANDORI *et al.*, 1998; SPARLA *et al.*, 1999).

22- O domínio BBE consiste de uma região na enzima Berberine Bridge Enzyme (enzBBE). A enzBBE consiste de um domínio FAD ligado covalentemente e um domínio de interação com o substrato, apresentando semelhanças com enzimas da família AA7 (Figura 18) (HUANG *et al.*, 2008; KUTCHAN *et al.*, 1995; WINKLER *et al.*, 2008). O domínio BBE na estrutura 3D2D compreende o domínio terminal da sequência que ajuda a interagir com o FAD e é próxima da região de interação e catálise do substrato. Foi realizado alinhamento entre as duas sequências BBE e duas sequências AA7. A região de domínio BBE apresentou vários resíduos conservados, com um tamanho de domínio próximos entre as sequências. O domínio das sequencias foi encontrado utilizando o Pfam com o mesmo procedimento realizado para as famílias do CAZy analisadas (WINKLER *et al.,* 2008).

```
3D2D_beberine   LLPVPEKVTVFRVTKNV-AIDEATSLLHKWQFVAEEL---EEDFTLSVLGGADEKQVWLT
3FW9_beberine   LLPVPEKVTVFRVTKNV-AIDEATSLLHKWQFVAEEL---EEDFTLSVLGGADEKQVWLT
1ZR6_glucose_ox TFEAPEIITTYQVTTTW-NRKQHVAGLKALQDWAQNTMPRELSMRL-EINANA-----LN
3RJA_carbohydra TFPAPKVLTRFGVTLNWKNKTSALKGIEAVEDYARWVAPREVNFRIGDYGAGN-----PG
                 : .*: :* : ** .     .    :.  :  *.     * .: :   ..

3D2D_beberine   MLGFHFGLKTVAKSTFDLLFPEL--GLVEEDYLEMSWGESFAYLAGLETVSQLNNRFL-K
3FW9_beberine   MLGFHFGLKTVAKSTFDLLFPEL--GLVEEDYLEMSWGESFAYLAGLETVSQLNNRFL-K
1ZR6_glucose_ox WEGNFFGNAKDLKKILQPIMKKAGGKSTISKLVETDWYGQINTYLYGADLNITYNYDVHE
3RJA_carbohydra IEGLYYGTPEQWRAAFQPLLDTLPAGYVVNPTTSLNWIESVLSYSNFDHVDFITPQPV-E
                  * .:*     :  :: ::       . .   . .*  ..        :.      : :

3D2D_beberine   FDERAFKTKVDLTKEPLPSKAFYGL-LERLSKEPNGFIALNGFGG---QMSKISSDFTPF
3FW9_beberine   FDERAFKTKVDLTKEPLPSKAFYGL-LERLSKEPNGFIALNGFGG---QMSKISSDFTPF
1ZR6_glucose_ox YFYANSLTAPRLSDEAIQAFVDYKFDNSSVRPGRGWWIQWDFHGGKNSALAAVSNDETAY
3RJA_carbohydra NFYAKSLTLKSIKGDAVKNFVDYYFDVSNKVKDRFWFYQLDVHGGKNSQVTKVTNAETAY
                       *   :. :.:   . * :  .        :   : .**    :: ::.  *.:

3D2D_beberine   PHRSGTRLMVEYIVAWNQSEQKKKTEFLDWLEKVYEFMKPFVSKNPRLGYVNHIDLDLGG
3FW9_beberine   PHRSGTRLMVEYIVAWNQSEQKKKTEFLDWLEKVYEFMKPFVSKNPRLGYVNHIDLDLGG
1ZR6_glucose_ox AHRDQLWLWQFYDSIYDYENNTSP-----YPESGFEFMQ---------GFVATIEDTLPE
3RJA_carbohydra PHRDKLWLIQFYDR---YDNNQT------YPETSFKFLD---------GWVNSVTKALPK
                .**.   *   *      .:: .      : *. ::*:.         *:*  :   *

3D2D_beberine   IDWGNKTVVNNAIEISRSWGESYFLSNYERLIRAKTLIDPNNVFNHPQSIPPMANFDYLE
3FW9_beberine   IDWGNKTVVNNAIEISRSWGESYFLSNYERLIRAKTLIDPNNVFNHPQSIPPMA------
1ZR6_glucose_ox DRKGKYFNYADTTLTKEEAQKLYWRGNLEKLQAIKAKYDPEDVFGNVVSVEPIA---YLE
3RJA_carbohydra SDWGMYINYADPRMDRDYATKVYYGENLARLQKLKAKFDPTDRFYYPQAVRPVK------
                   *     :.        : *:  *  :*   *:  ** : *    :: *:

3D2D_beberine   KTLGS--DGGEVVI------
3FW9_beberine   --------------------
1ZR6_glucose_ox QKLISEEDLNSAVDHHHHHH
3RJA_carbohydra --------------------
```

Figura 18 - Alinhamento entre membros da enzBBE e membros da família AA7. Alinhamento entre as sequências berberine 3D2D e 3FW9 com as sequências das enzimas da família AA7, 1ZR6 glucooligosaccharide oxidase e 3RJA carbohydrate oxidase. A região destacada foi identificada como domínio BBE pelo Pfam.

23- O domínio Amidohydro_2 pertence a superfamília hidrolase dependente de metal. Essa estrutura está envolvida na região catalítica da urease de *Kiebsiellaaerogenes*, coordenando um átomo de níquel e apresenta similaridade conformacional com a adenosinedeaminase, coordenando um átomo de zinco (JABRI *et al.*, 1995). Existe uma semelhança entre o domínio Amidohydro_2 com regiões de várias outras proteínas, podendo estar envolvido em catálise ou não, sendo que algumas sequências apresentam perda do local de interação com o metal (HOLM *et al.*, 1997).

24- A Glyco_hydro_61 consiste em uma família de enzimas que apresenta atividade endoglucanase fraca, conhecidas como glycosidehydrolases 61. Recentemente percebeu-se que se tratava de lyticpolysaccharidemonooxygenases, sendo reclassificado pelo Cazy como família AA9 (HARRIS *et al.*, 2010; LEVASSEUR *et al.*, 2013). As enzimas que constituem essa família são capazes de aumentar a atividade de outras glicosil hidrolases (HARRIS *et al.*, 2010).

25- O domínio Lactamase_B_2 é encontrado em enzimas da superfamília beta-lactamase e está relacionada com a dobra de proteínas metalo-beta-lactamase. A enzima metalo-beta-lactamase está envolvida na resistência a antibióticos em bactérias (CARFI *et al.*, 1995).

27- O domínio CFEM é uma estrutura de cerca de 60 aminoácidos contendo 8 resíduos de cisteína e é encontrado em proteínas de membrana fúngicas (CARFI *et al.*, 1995). Esse domínio está envolvido na patogenicidade de fungos, sendo encontrado em cerca de 4000 sequências de fungos. O domínio CFEM está envolvida na aquisição de heme pelos fungos patogênicos, é encontrado em *Candidaalbicans*, apresentando papel de extração da heme da hemoglobina (CARFI *et al.*, 1995).

28- O domínio LPMO_10 consiste em um domínio associado a uma grande variedade de domínios de ligação à celulose. Esse domínio é encontrado na enzima da bactéria *Streptomycesolivaceoviridis*, com número de acesso CAA55284.1 no banco de dados EMBL. A sequência CAA55284.1 foi descrita tendo uma região com aminoácidos aromáticos com evidências de interação com o substrato quitina. Esses aminoácidos estão dentro do domínio LPMO_10, descrito para a sequência CAA55284.1 no Pfam. A localização de domínio da sequência CAA55284.1 é descrita no Uniprot para o código Q54501 (FORSBERG *et al.*, 2014; SCHNELLMANN *et al.* 1994).

29- A SGL é um domínio presente em gluconolactonase de *Zymomonasmobilis* capaz de oxidar D-glucono-1,5-lactone em D-gluconate. O domínio está presente na sequência da gluconolactonase CAA47637.1 no intervalo 74-345, sendo que a enzima contém 356 aminoácidos. Esses dados estão disponíveis no Uniprot Q01578 (KANAGASUNDARAM *et al.*, 1992). O domínio SGL também é encontrado na enzima humana SMP30. Analisando a estrutura de uma SMP30, PDB 3G4E descrita no Uniprot Q15493, pode-se ver que a enzima

apresenta o domínio SGL, podendo ter funções de oxidação do D-glucono-1,5-lactone. O domínio foi encontrado da posição 16 à 264, sendo que a enzima completa tinha 299 aminoácidos, abrangendo grande parte dos aminoácidos previstos como sítio de ligação e catálise (AIZAWA *et al.*, 2013). A proteína SMP30 apresenta atividade enzimática promíscua, sendo capaz de se ligar a diversos metais divalentes podendo influenciar na atividade da enzima dependendo do metal ligado (DUTTA *et al.,* 2019).

30- O domínio GSDH é encontrado em glucose dehydrogenase dependente de pirroloquinolinaquinona que oxida a glicose em gluconolactona. Na glucose dehydrogenase de *Acinetobactercalcoaceticus*, constituída pelo modelo PDB 1CQ1, é encontrado um domínio GSDH como mostrado no Uniprot pelo código P13650 (OUBRIE, 1999). Analisando os dados do Pfam no banco de dados do Uniprot para a P13650, pode-se notar que a região de ligação do PQQ corresponde ao local identificado como domínio GSDH na estrutura 1CQ1 e provavelmente esse domínio constitui uma dobra de ligação ao PQQ (OUBRIE, 1999).

Apêndice G - Revisão da literatura avaliando os domínios e trabalhos estruturais

Domínios encontrados na literatura para a família AA3sub1

A enzima CDH foi descoberta pela primeira vez por UllaWestermark e Karl-Erik Eriksson, sendo uma enzima denominada celobiosequinonaoxidoredutase (CBQ), contendo um grupo flavina (WESTERMARK *et al.*, 1974; WESTERMARK *et al.*, 1975). Posteriormente foi purificada uma forma contendo flavina e heme, a qual foi denominada de celobiose oxidase (CBO). Posteriormente, identificaram que a 'CBQ' era um fragmento ativo catalítico de 'CBO', sendo a 'CBO' renomeada para CDH (WOOD *et al.*, 1992).

A enzima CDH pode ser encontrada na natureza contendo ou não o domínio HEME, tendo atividades distintas quanto ao substrato receptor de elétrons (SAMEJIMA *et al.*, 1992). Foi descrito também que os domínio independentes de HEME e flavina isolados de culturas fúngicas não foram capazes de interagir de forma que a flavina reduzida pudesse reduzir heme, no entanto, a atividade foi verificada quando a enzima que continha o domínio heme era clivada por proteólise de papaína.

Domínios encontrados na literatura para a família AA3sub2

O grupo AA3sub2 é constituído pela aril álcool oxidase and glucose 1-oxidase. Ambas as enzimas consistem de um domínio GMC_oxred_N de ligação do FAD e um domínio GMC_oxred_C de ligação do substrato (BANKAR *et al.*, 2009; HERNÁNDEZ-ORTEGA *et al*., 2012; UNIPROT, 2019b, 2019c). Dados de filogenia mostram que as sequências de GOX e AOX ficam bem próximas, pertencendo ao mesmo ramo e dentro de um clado com outras GMC correlacionadas (ZÁMOCKÝ *et al.*, 2004).

Domínios encontrados na literatura para a família AA3sub4

A família AA3sub4 é formada pela Piranose oxidase, ela faz parte da família de enzimas Glucose-methanol-cholineoxidoreductase, tendo como característica pela existência de 2 domínios GMC_oxred_N de interação ao FAD e GMC_oxred_C de interação com o substrato (ALBRECHT *et al.*, 2003; HERZOG *et al.*, 2019). Dados de alinhamento de sequência e de predição de estrutura secundária mostra uma baixa conservação de sequência,

mas com conservação de características estruturais preditas (ALBRECHT *et al.*, 2003; SÜTZL *et al.*, 2019).

Domínios encontrados na literatura para a família AA4

A família AA4 é formada pela enzima vanilil-álcool oxidases, sendo um dos membros da família de oxidoredutases VAO / PCMH, compartilham um domínio de ligação ao FAD, sendo conservado na família enzimática e apresentando um domínio de ligação ao substrato que é variável (EWING *et al.*, 2016). O domínio de interação com o FAD se divide em dois intervalos como no caso da 1VAO, que se encontra no intervalo 6-270 e 500-560. Na 1VAO, o domínio de interação ao substrato consiste nos resíduos 271–499, cobrindo o anel isoaloxazina da FAD (MATTEVI *et al.*, 1997).

Domínios encontrados na literatura para a família AA5

Não foi encontrado estrutura ou trabalhos referentes aos domínios da Glioxal oxidase, porém existe grande associação da sua sequência com as sequências da galactose oxidase, mostrando conformação estrutural parecida e aminoácidos catalíticos conservados (MATTEVI *et al.*, 1997). O trabalho de Firbank mostra que a galactose oxidase apresenta três domínios, sendo o segundo consistindo de atividade funcional e contendo o sítio catalítico. Não se sabe ao certo a função do terceiro domínio, porém parte de sua estrutura participa na ligação ao cobre (FIRBANK *et al.*, 2001). No trabalho de Yin, descreve-se uma álcool oxidase que apresenta apenas dois domínios: o primeiro domínio correspondendo a um conjunto de estruturas Kelch, que juntas formam uma β-hélice clássica de sete pás, correspondendo provavelmente aos domínios II, e o segundo domínio provavelmente corresponde ao domínio III do trabalho de Firbank, visto que são os domínios de interação com o cobre. Yin mostra que o domínio faltante no álcool oxidases corresponde ao domínio CBM32 N-terminal, o qual provavelmente não participa da catálise (YIN *et al.*, 2015).

Domínios encontrados na literatura para a família AA6

As enzimas 1,4-benzoquinone reductases constituem a família AA6 do CAZy. Ela consiste de uma flavoproteína com estrutura semelhante às flavodoxinas. É visto que a NADH

e a quinona se ligam a regiões semelhantes do sítio ativo, próximas ao anel isoaloxazina do cofator FMN. Aparentemente a enzima é composta por um único domínio e sua estrutura funcional consiste de oligômeros (ANDRADE *et al.*, 2007; KOCH *et al.*, 2017).

Domínios encontrados na literatura para a família AA7

As enzimas que constituem a família AA7 apresentam dois domínios, um de ligação ao FAD e um de ligação ao substrato e pertence a superfamília PCMH. O domínio de ligação ao FAD é constituído por dois subdomínios um N-terminal e outro C-terminal. A enzima com estrutura no PDB 1ZR6, apresenta regiões de domínios constituídas pelo N-terminal do aminoácido 1-206 e os resíduos C-terminais 421-474. Entre os indivíduos da Superfamília PCMH existe uma diferença estrutural entre os domínios, sendo o domínio de ligação ao FAD muito mais sobreposto estruturalmente entre os membros da família PCMH doque o domínio de interação ao substrato (HUANG *et al.*, 2005).

Domínios encontrados na literatura para a família AA8

As enzimas que constituem a família AA8 consistem do domínio citocromo b. Essa sequência pode ser encontrada associada a enzima CDH ou a outros módulos enzimáticos (LEVASSEUR *et al.*, 2013).

Domínios encontrados na literatura para as LPMO

A Lyticpolysaccharidemonooxygenases constituem as enzimas das famílias AA9-AA11 e AA13-AA16, podendo ser encontradas na forma de domínios únicos ou associadas a CBM. Esses domínios são encontrados em bactérias, vírus, fungos, artrópodes e em plantas, (FORSBERG *et al.*, 2014; FRANDSEN *et al.*, 2019). Essa enzima apresenta uma região de interação com um íon de cobre, formadas por duas histidinas altamente conservadas (SIMMONS *et al.*, 2017; YIN *et al.*, 2015). As famílias AA9 e AA11 são constituídas comumente por enzimas fúngicas enquanto a família AA10 abrange vários outros grupos de organismos, sendo muito encontrada em bactérias decompositoras de celulose (FORSBERG *et al*., 2014). A família AA11 compartilha semelhanças com sequências das famílias AA9 e AA10, tendo estrutura tridimensional muito parecida também. O trabalho da estrutura PDB

4MAH relata a existência de um motivo estrutural extra que se assemelha a regiões em outras enzimas diversas (HEMSWORTH *et al.*, 2014).

As LPMO da família AA13 apresentam apenas um domínio, podendo aparecer associada ao domínio CBM20, apresentando atividade em amido. As enzimas desta família também podem apresentar uma semelhança com o domínio das enzimas AA10 (LO LEGGIO *et al.*, 2015).

A família AA14 foi identificada como uma enzima que degrada xilano. Sua caracterização foi dificultada devido a enzima ter atividade apenas em xilano adsorvido em celulose. Curiosamente, essa enzima teve um módulo CBM1 acrescentado em sua estrutura e mostrou uma diminuição em seu desempenho, mostrando que módulos de ligação talvez não sejam interessantes para o funcionamento da enzima. A estrutura da 5NO7, pertencente a uma enzima AA14, revelou diferenças em relação à região do sítio ativo, quando comparado com a família AA9 (COUTURIER *et al.*, 2018).

A família AA15 não foi encontrada em fungos. A família AA16 foi encontrada analisando genomas de fungos, mostrando similaridade com outras enzimas LPMO, como histidina metilada, ligação ao cobre e conservação em aminoácidos. Ela apresentou atividade em celulose, aumentando a eficiência de conversão de celulases quando o substrato era pré-tratado com AA16 ativas em celulose. As sequências encontradas no trabalho de caracterização dessa família indicam a existência de sequências contendo domínios CBM ou contendo âncoras glycosylphosphatidylinositol (FILIATRAULT-CHASTEL *et al.*, 2019) .

Domínios encontrados na literatura para a família AA12

As enzimas que constituem a família AA12 consistem de um domínio de ligação ao pirroloquinolinaquinona. Podem ser encontradas associadas aos domínios CBM1 ou citocromo b, sendo queasenzimas com os três domínios são mais raras de se encontrar. A estrutura de um membro da família AA12, PDB 6H7T mostrou grande similaridade de arquitetura global com a glucose dehydrogenase de *Acinetobactercalcoaceticus*(PDB 1CQ1). Uma análise filogenética de 59 sequências identificadas como prováveis AA12, mostrou que essas sequências se dividem em três subgrupos, com dois grupos relacionados por taxonomia, sendo um Ascomycota e outro Basidiomycota. Um terceiro grupo é formadopor representantes dessesdois filos (TURBE-DOAN *et al.*, 2019).

Apêndice H - Gráficos gerados na análise

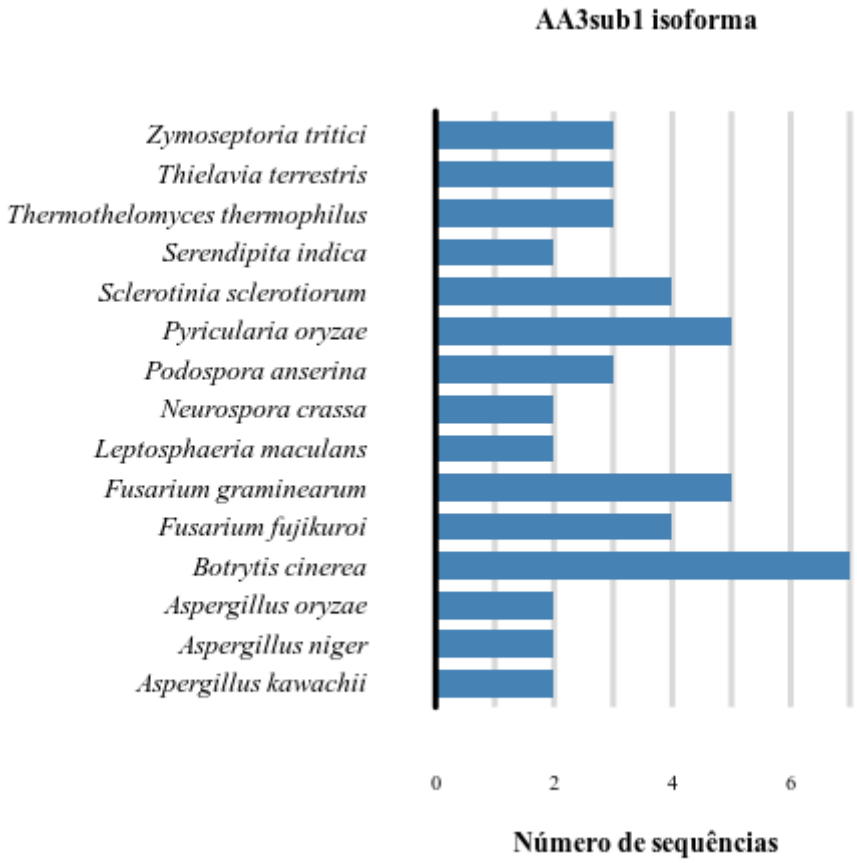

Figura 19 - Gráfico do número de isoformas da subfamília AA3sub1. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

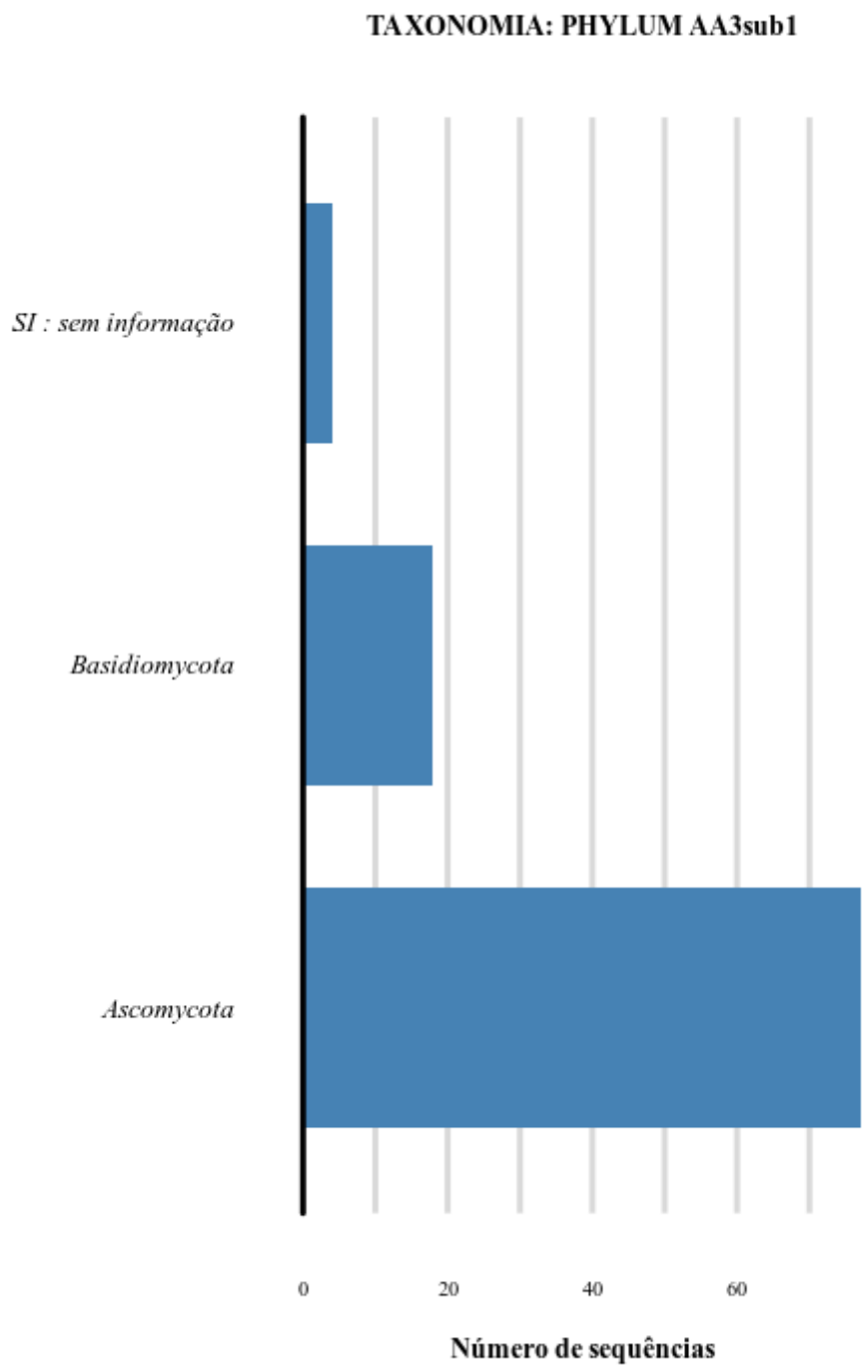

Figura 20 - Gráfico do número de sequências dentro dos filos da subfamília AA3sub1. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome dos filos e no eixo x o número de sequências.

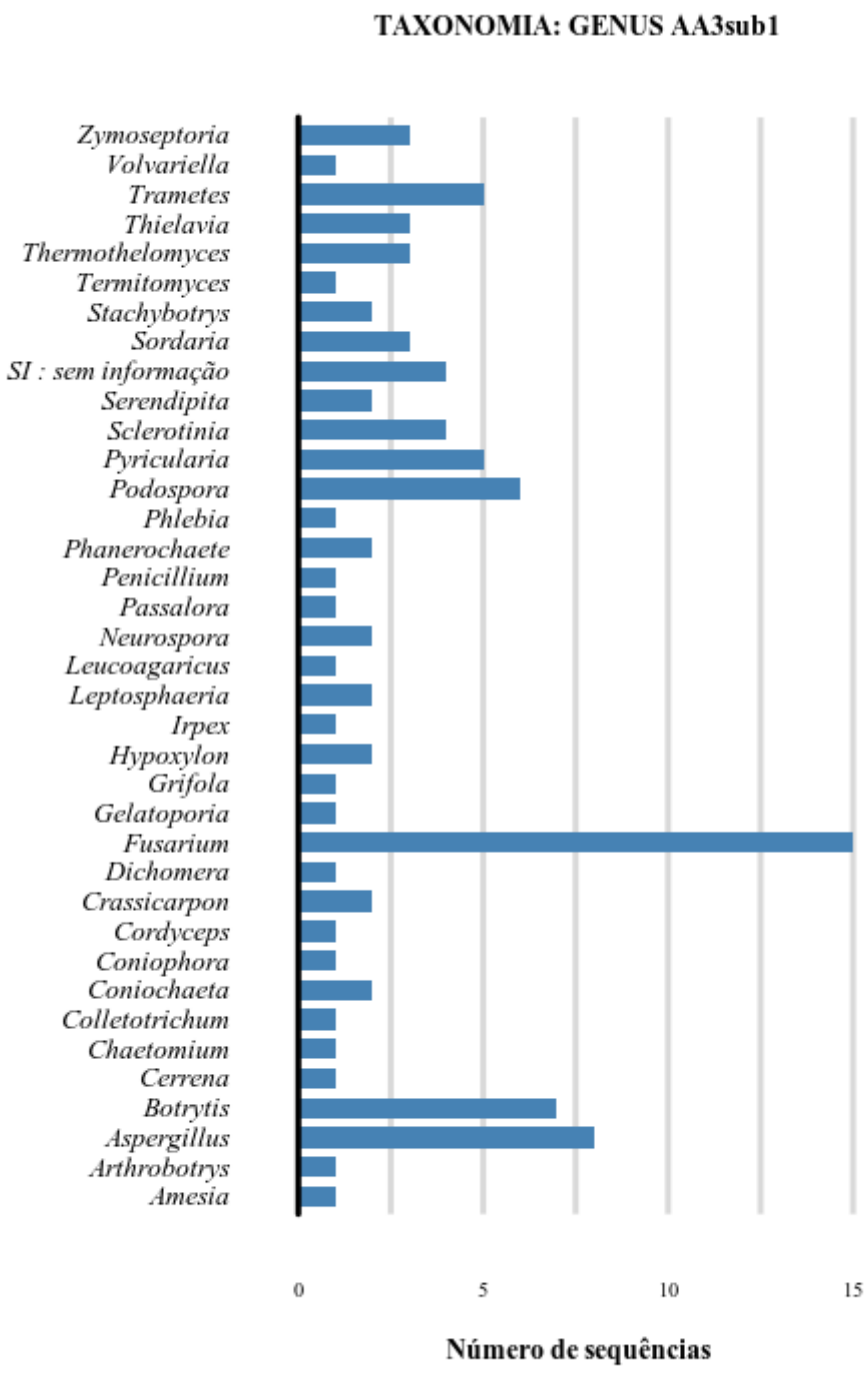

Figura 21 - Gráfico do número de sequências dentro dos gêneros da subfamília AA3sub1. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome dos gêneros e no eixo x, o número de sequências.

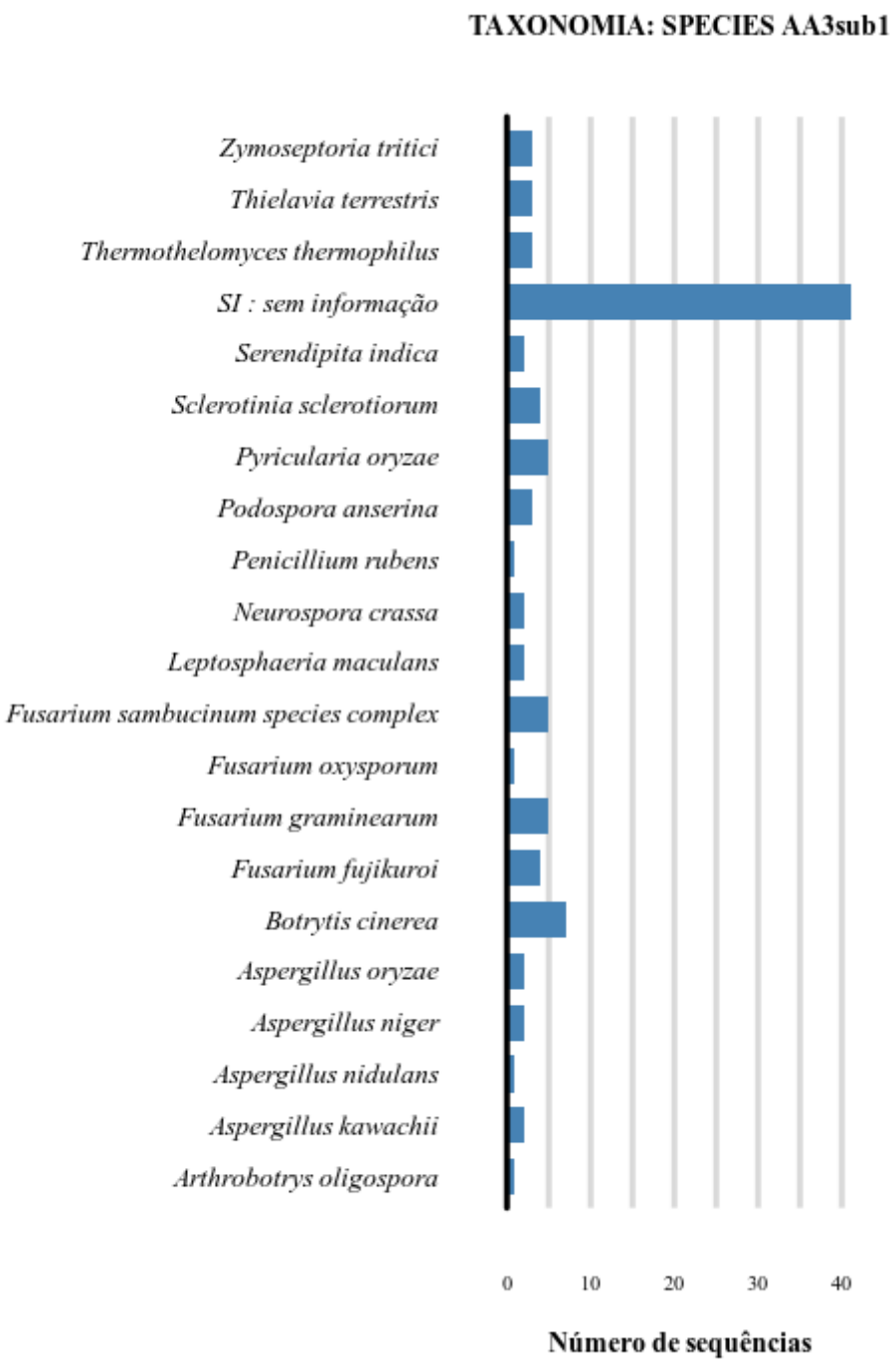

Figura 22 - Gráfico do número de sequências dentro das espécies da subfamília AA3sub1. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

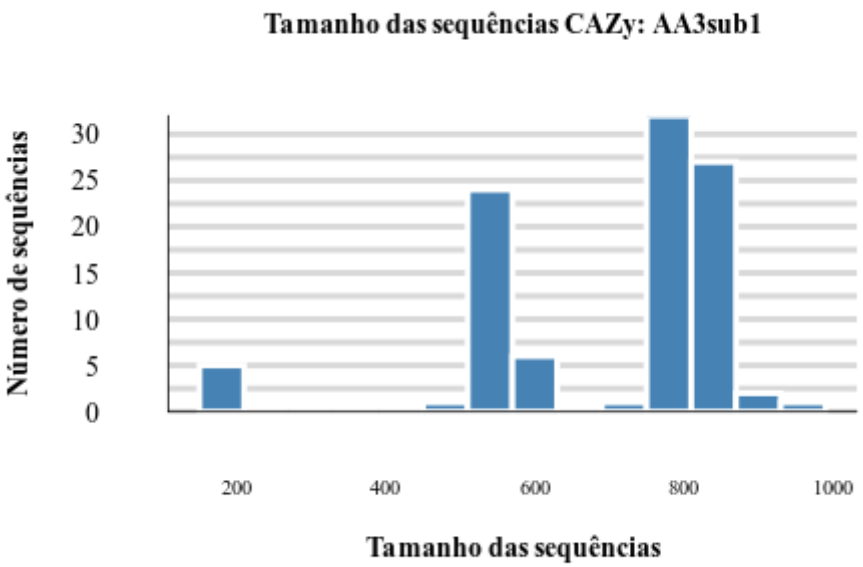

Figura 23 - Gráfico de tamanho de sequência da subfamília AA3sub1. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

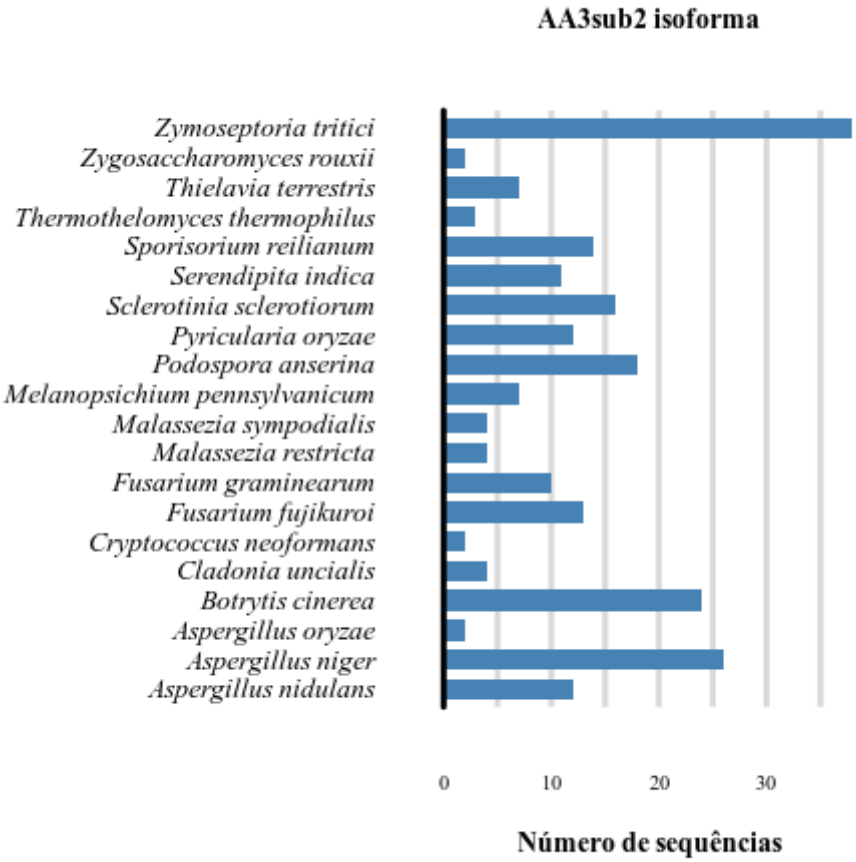

Figura 24 - Gráfico do número de isoformas da subfamília AA3sub2. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

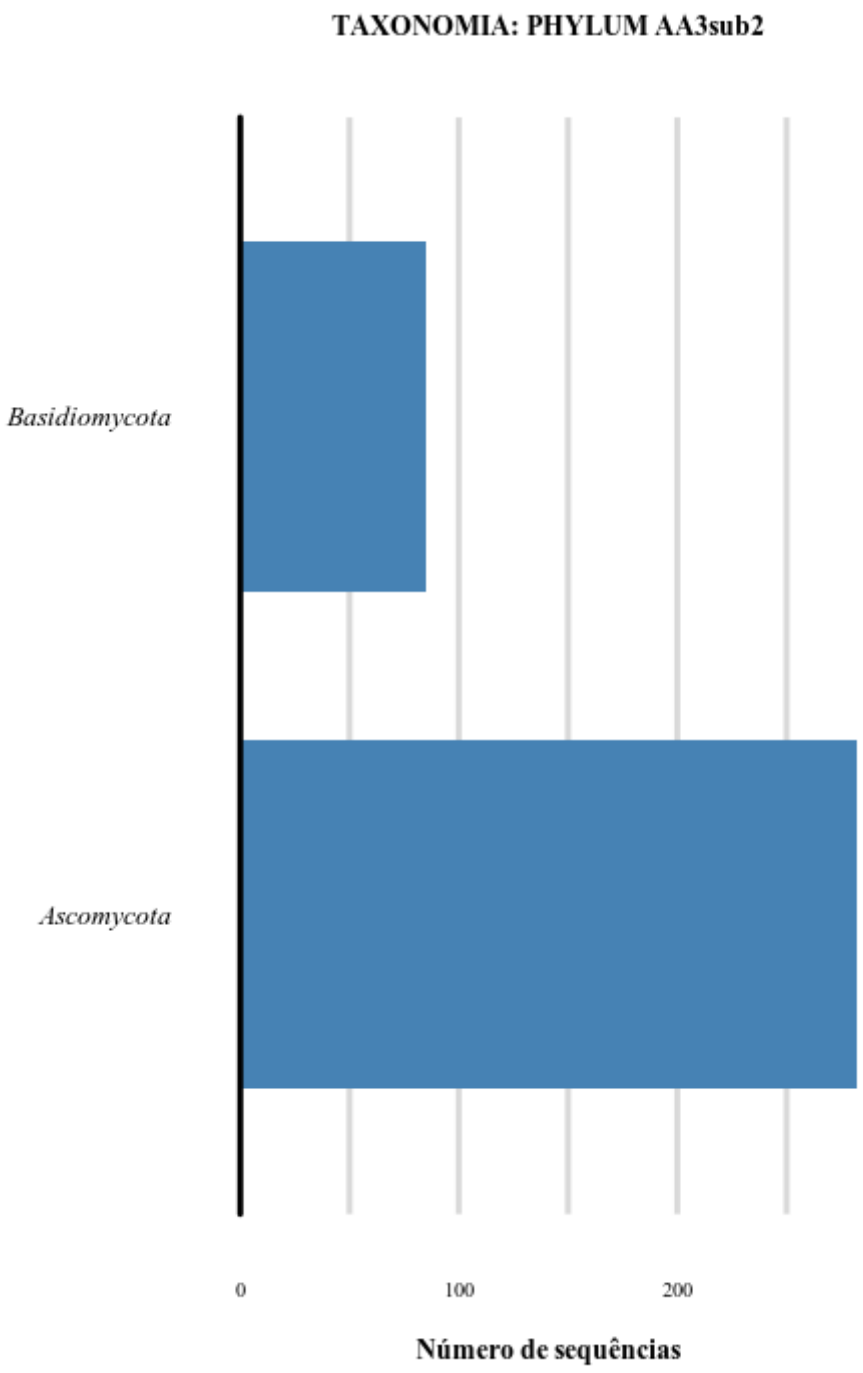

Figura 25 - Gráfico do número de sequências dentro dos filos da subfamília AA3sub2. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

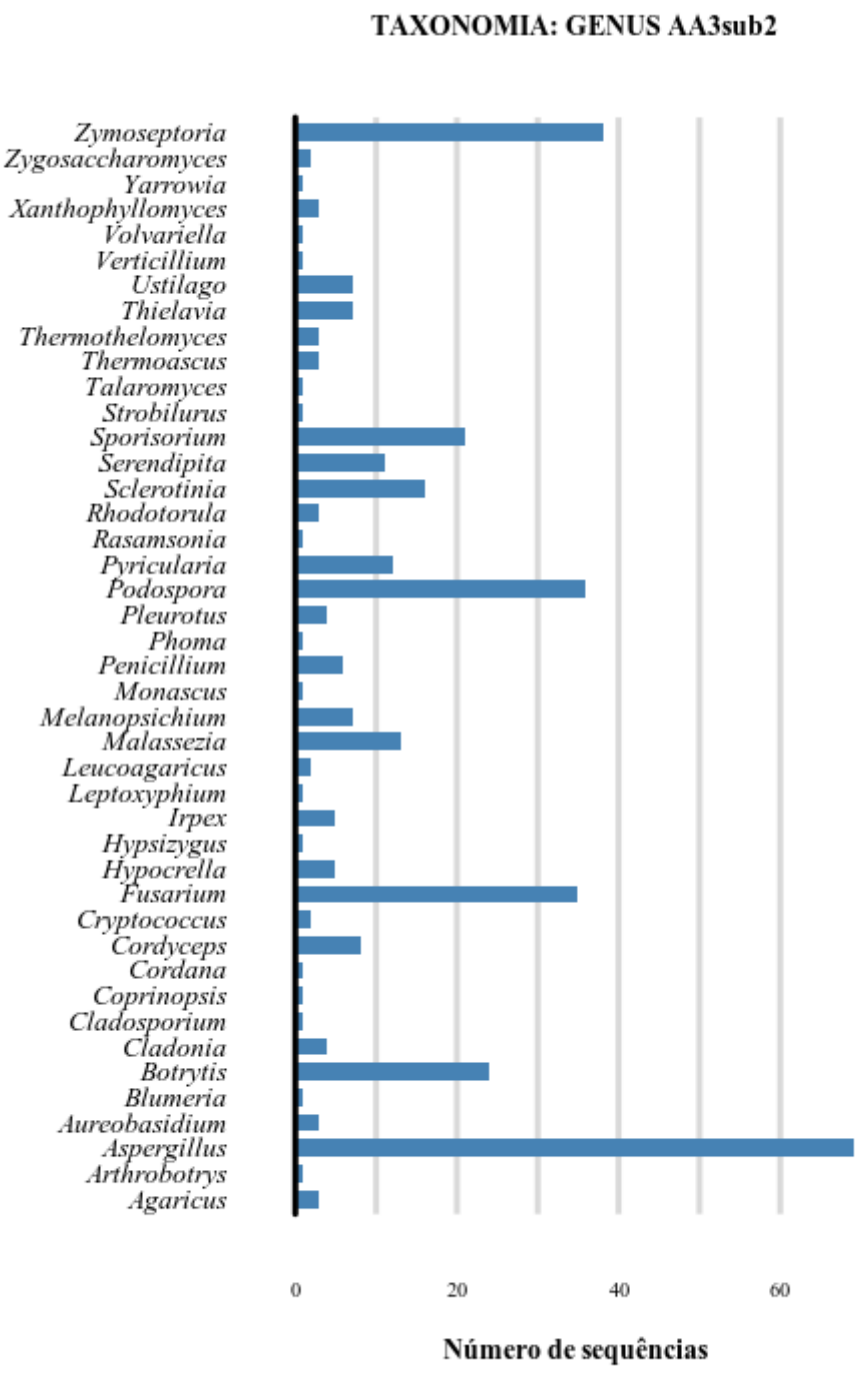

Figura 26 - Gráfico do número de sequências dentro dos gêneros da subfamília AA3sub2. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

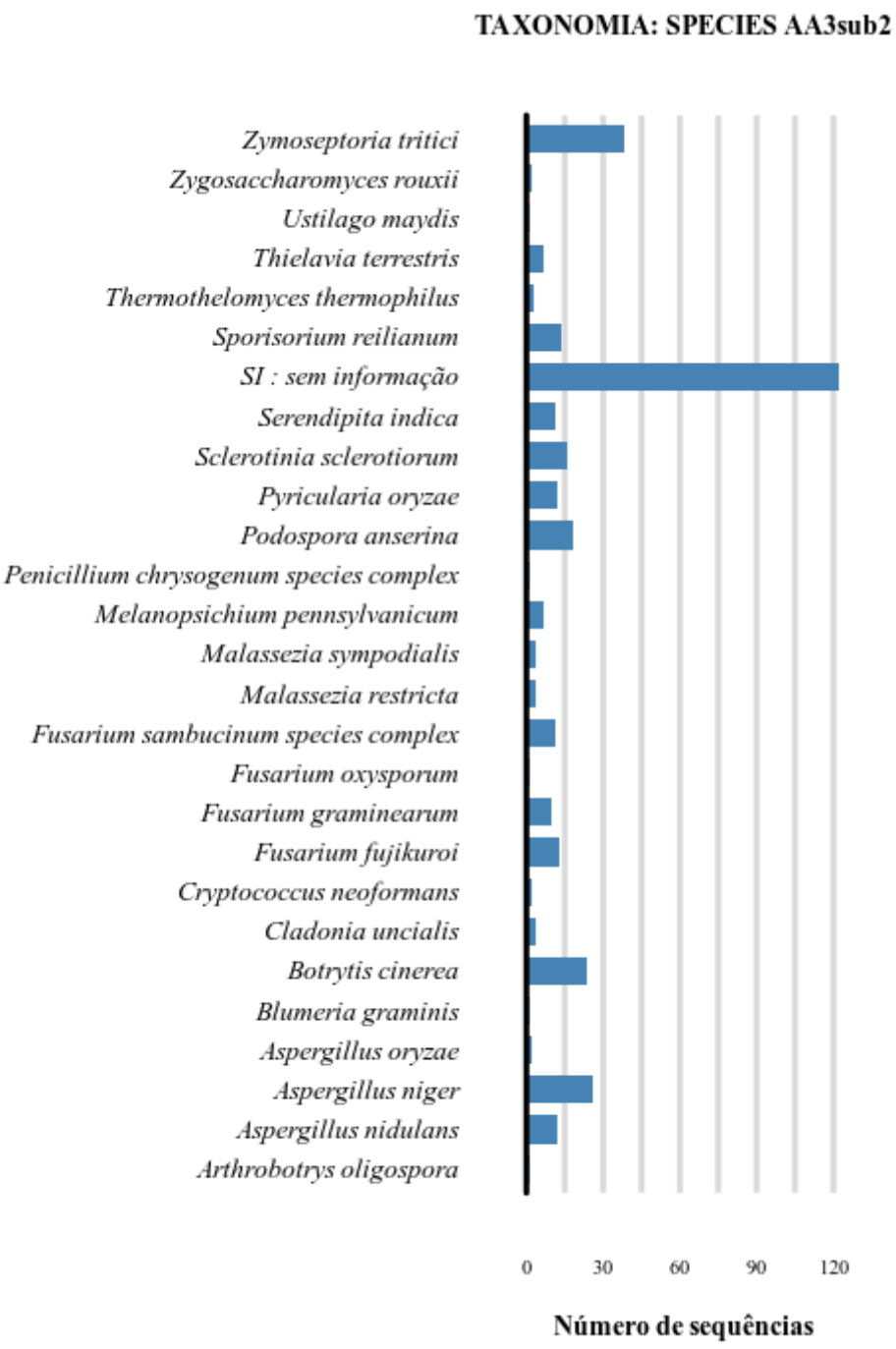

Figura 27 - Gráfico do número de sequências dentro das espécies AA3sub2. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

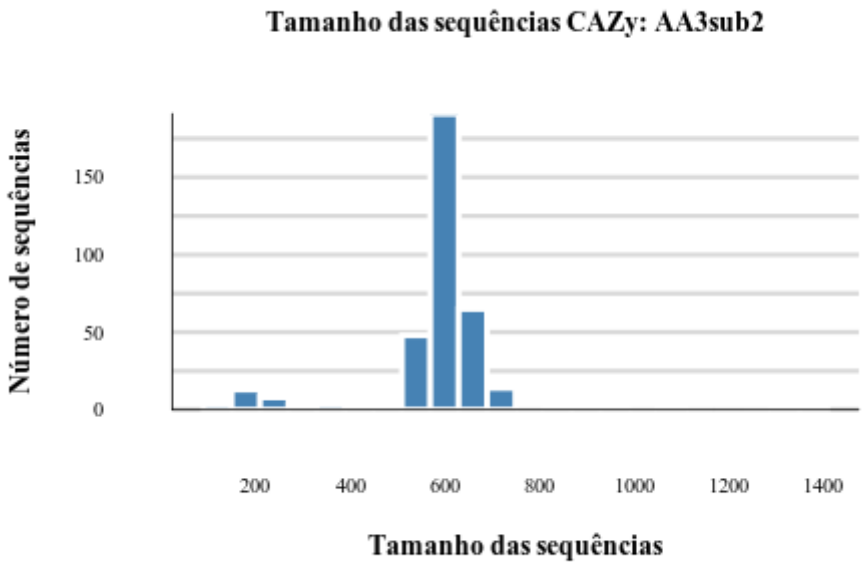

Figura 278 - Gráfico de tamanho de sequências da subfamília AA3sub2. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

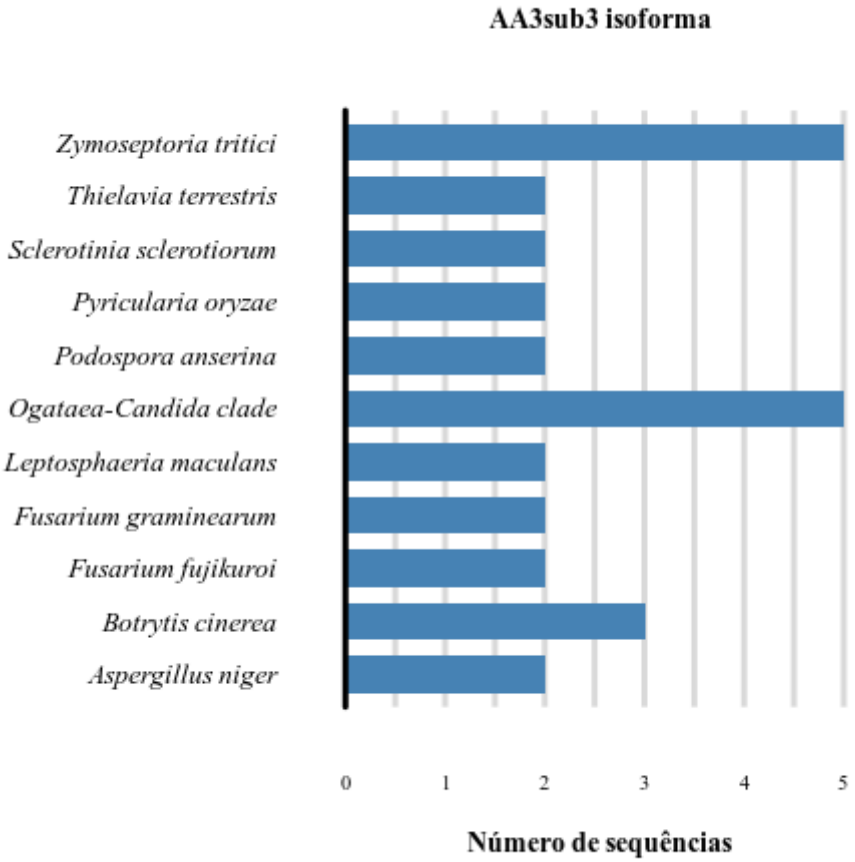

Figura 29 - Gráfico do número de isoformas da subfamília AA3sub3. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

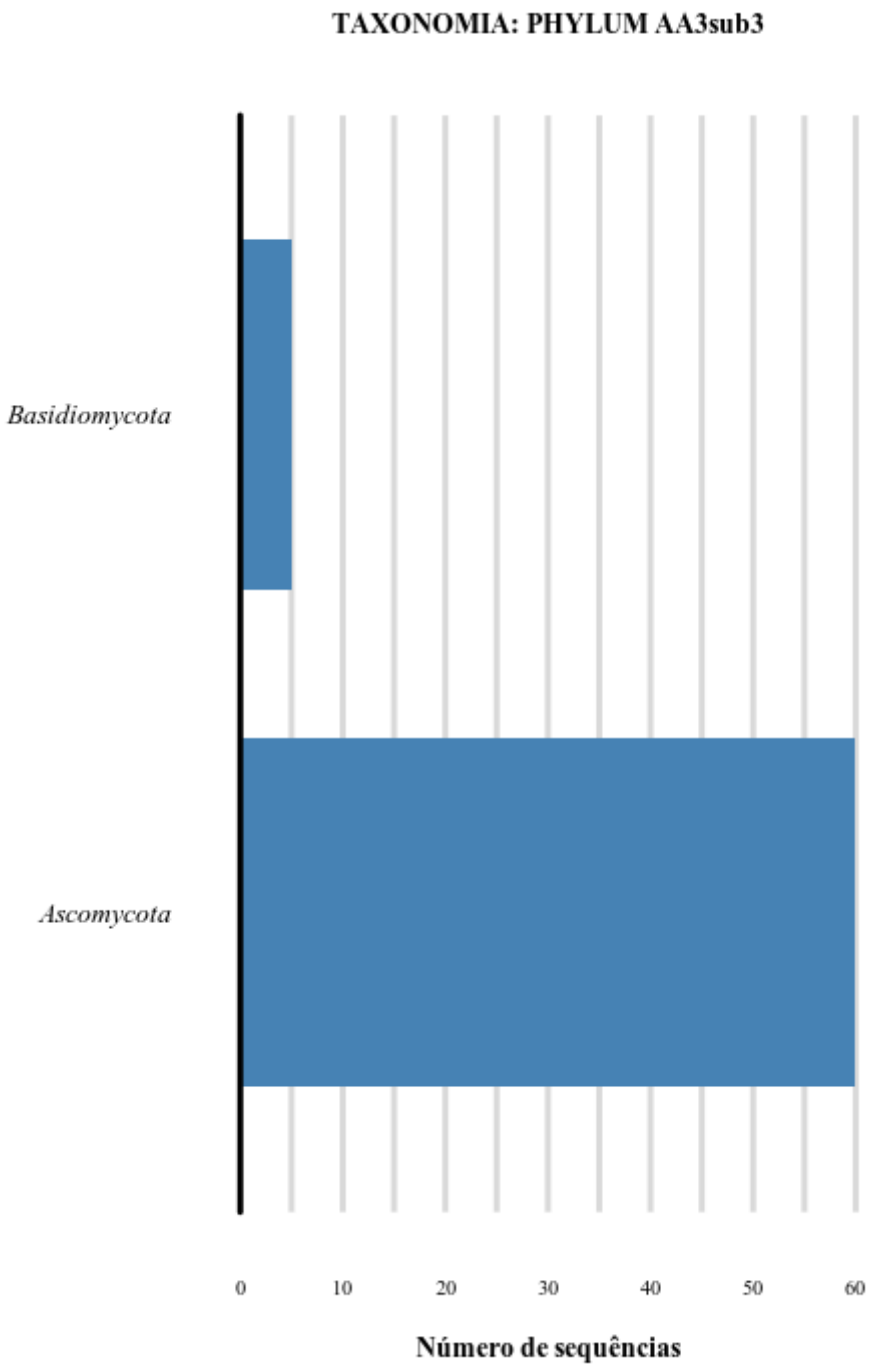

Figura 30 - Gráfico do número de sequências dentro dos filos AA3sub3. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome dos filos e no eixo x o número de sequências.

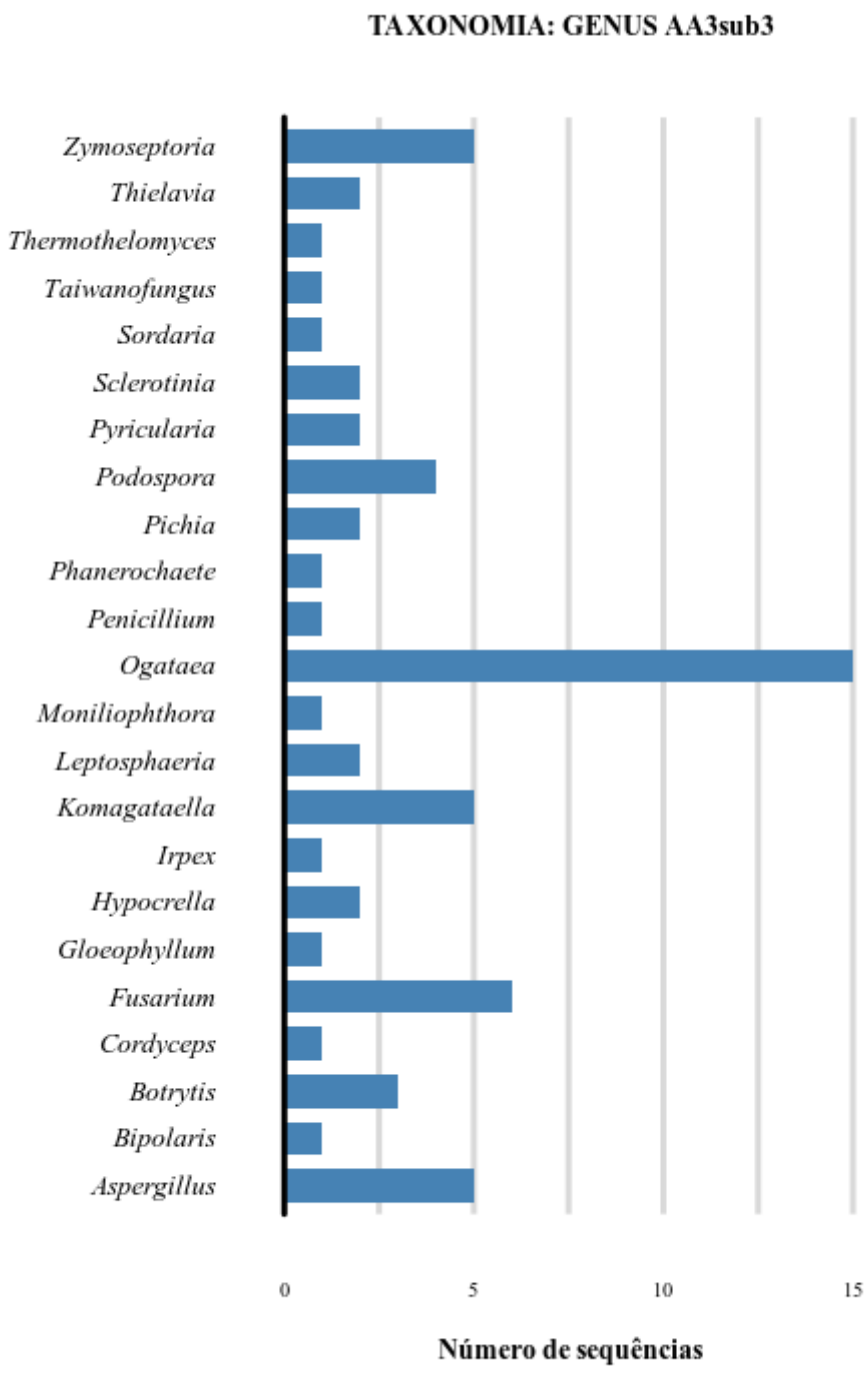

Figura 31 - Gráfico do número de sequências dentro dos gêneros da subfamília AA3sub3. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

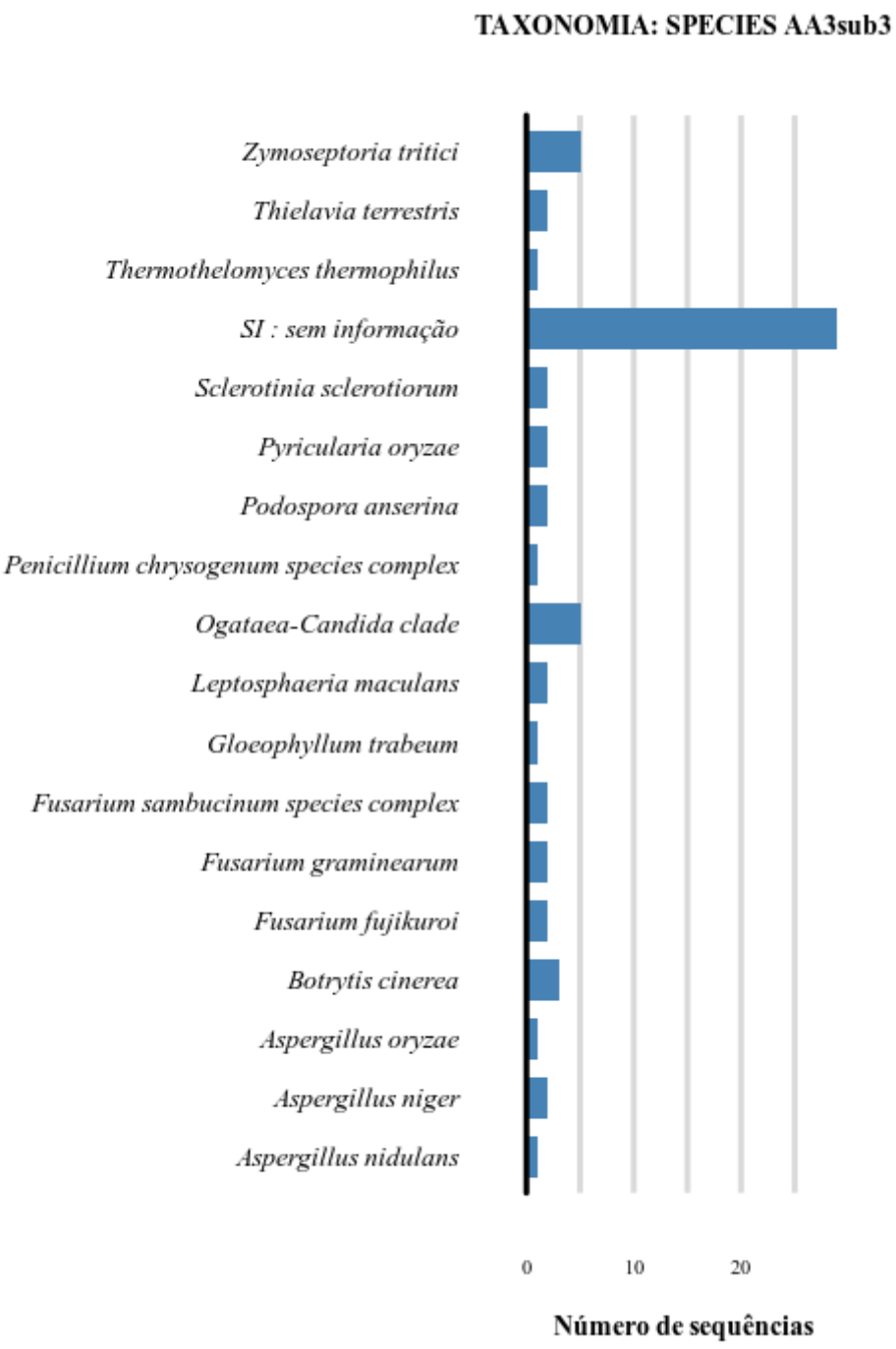

Figura 32 - Gráfico do número de sequências dentro das espécies da subfamília AA3sub3. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

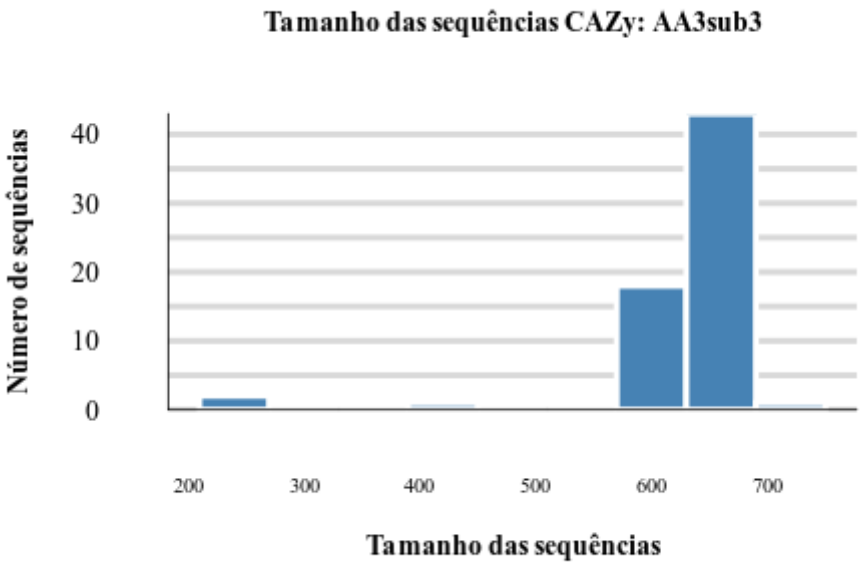

Figura 33 - Gráfico de tamanho de sequência AA3sub3. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

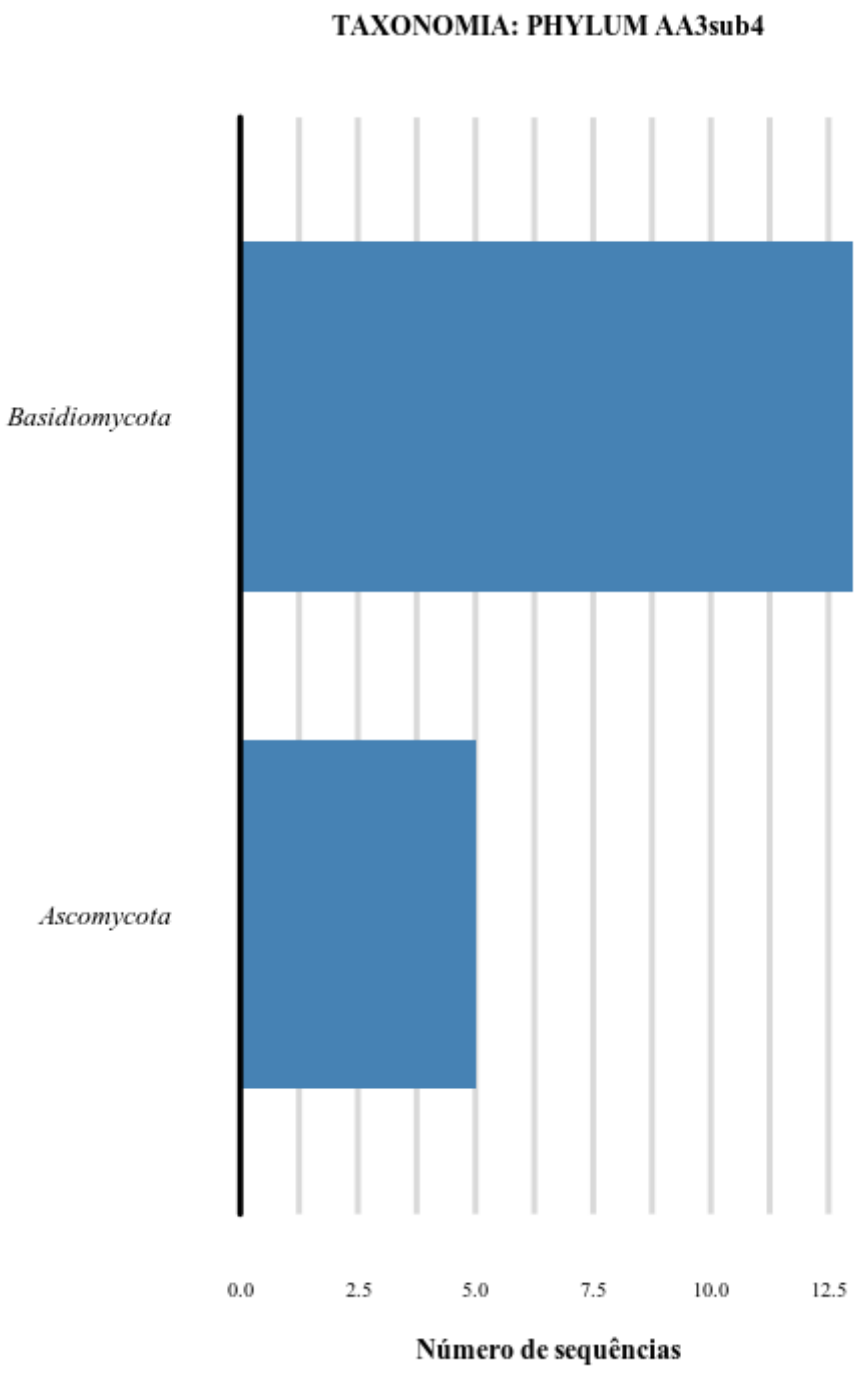

Figura 34 - Gráfico do número de sequências dentro dos filos da subfamília AA3sub4. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

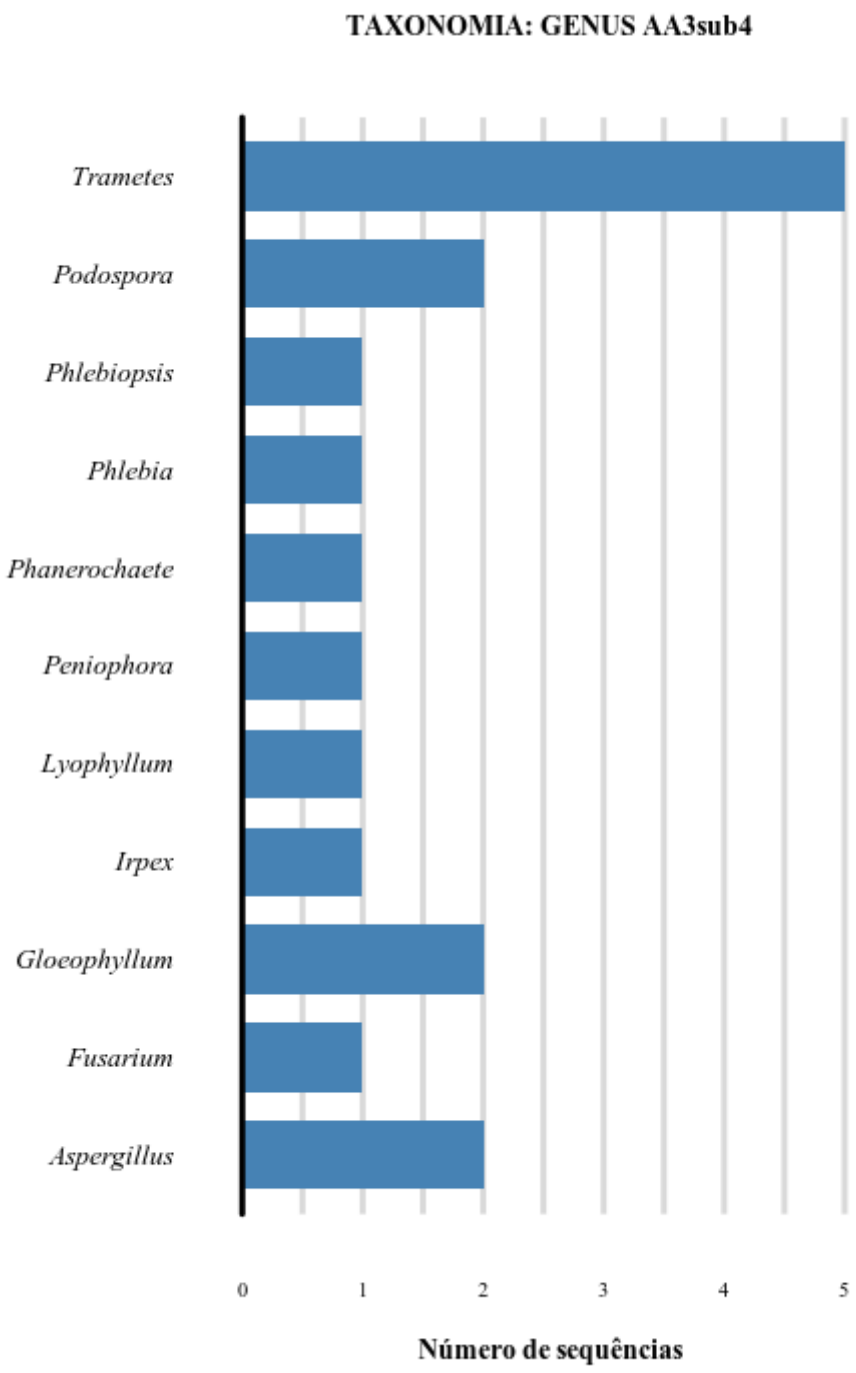

Figura 35 - Gráfico do número de sequências dentro dos gêneros da subfamília AA3sub4. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

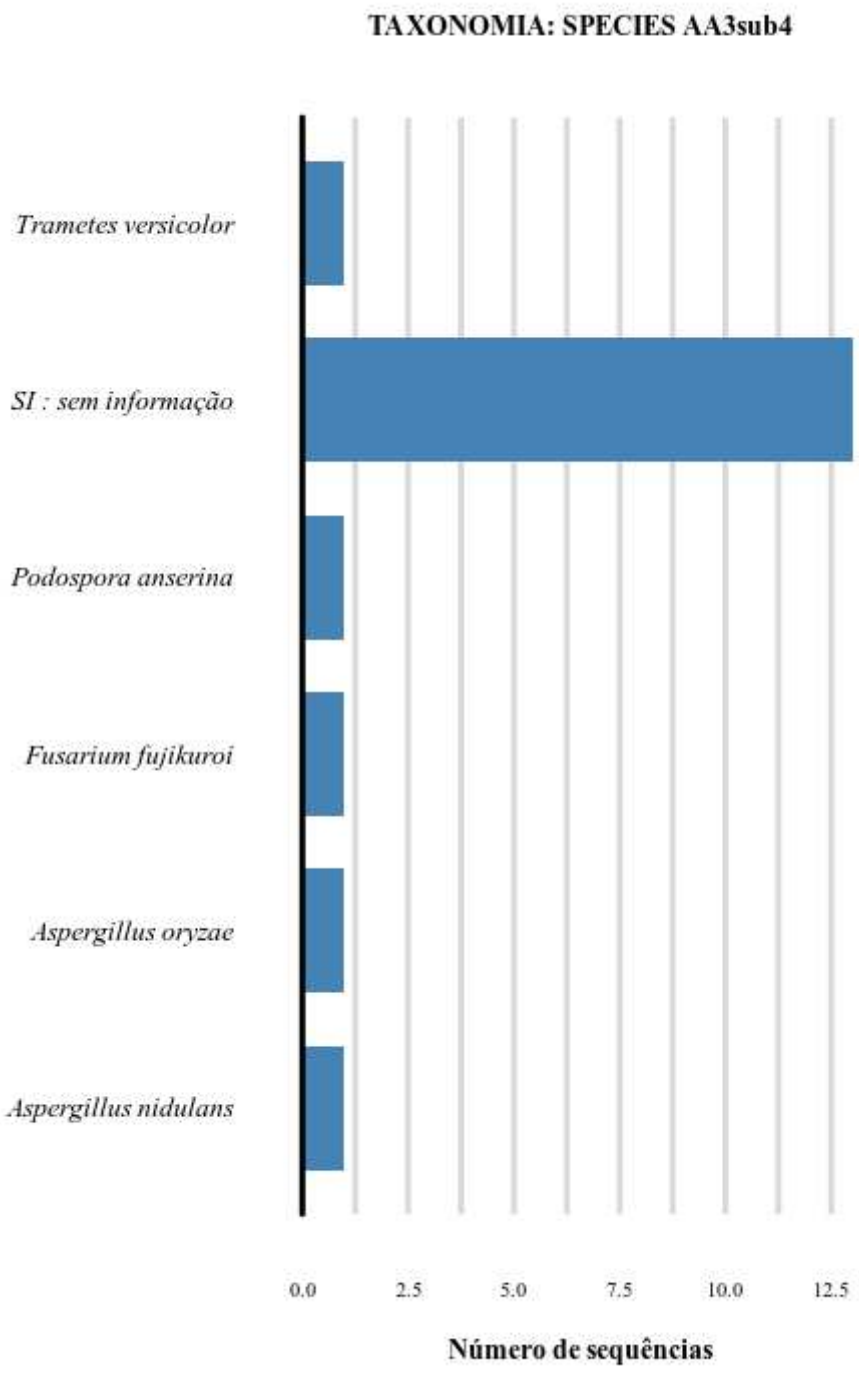

Figura 36 - Gráfico do número de sequências dentro das espécies da subfamília AA3sub4. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

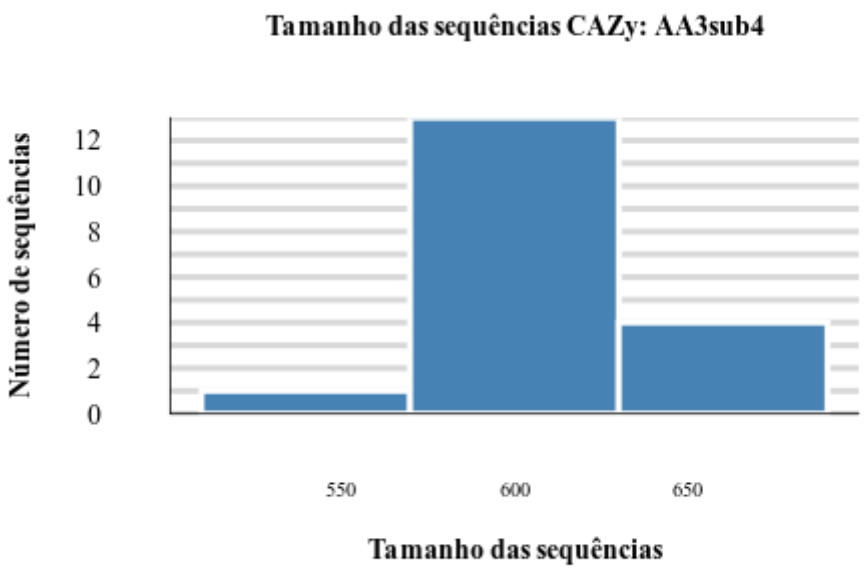

Figura 37 - Gráfico de tamanho de sequências da subfamília AA3sub4. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

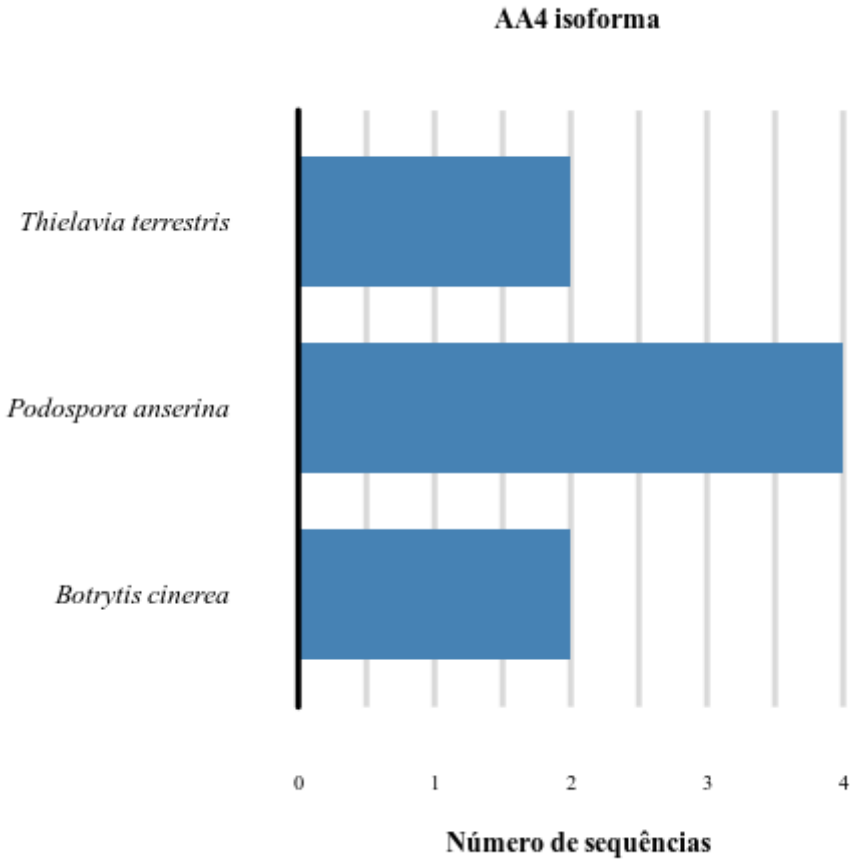

Figura 38 - Gráfico de isoforma da família AA4. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

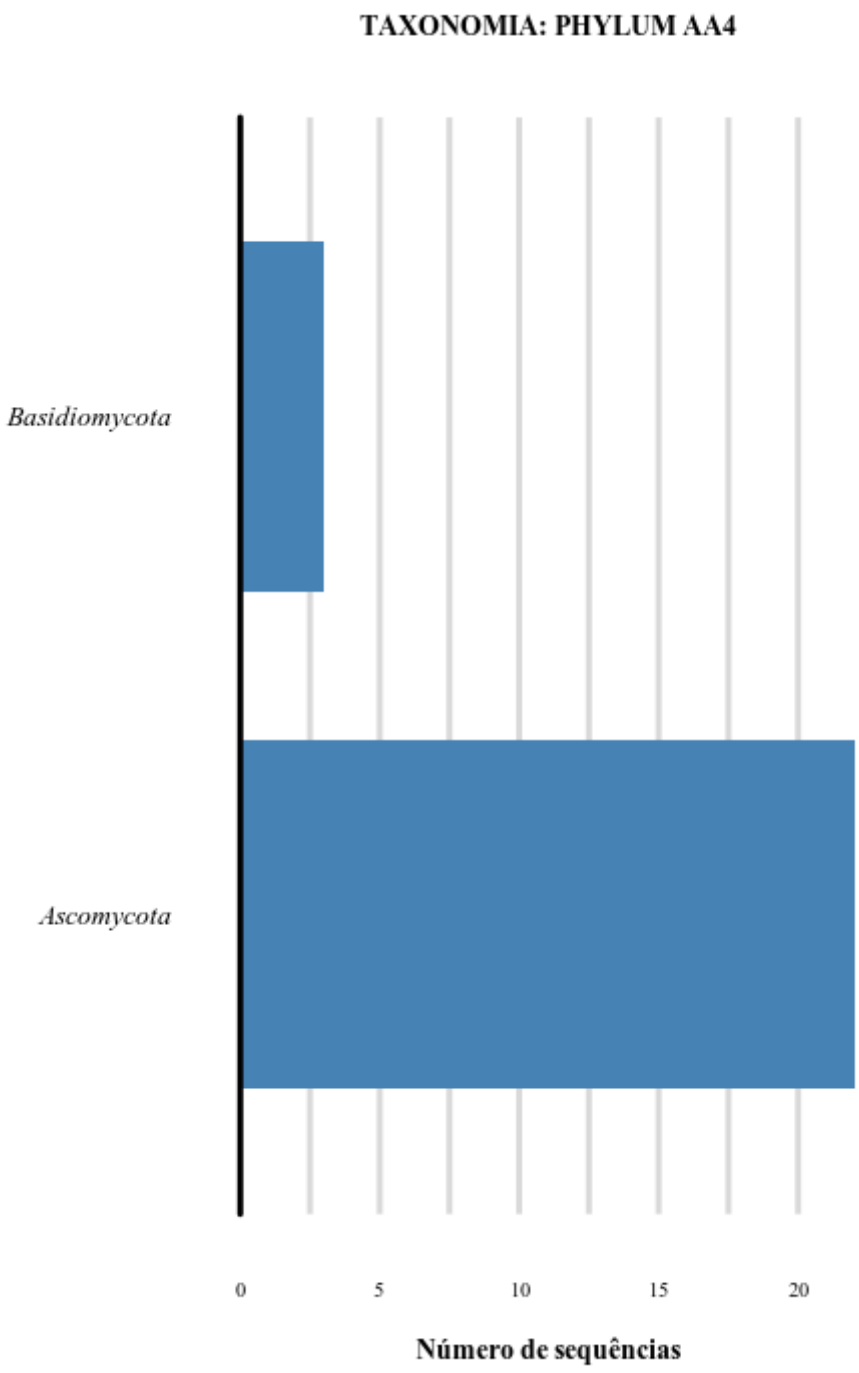

Figura 39 - Gráfico do número de sequências dentro dos filos da família AA4. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

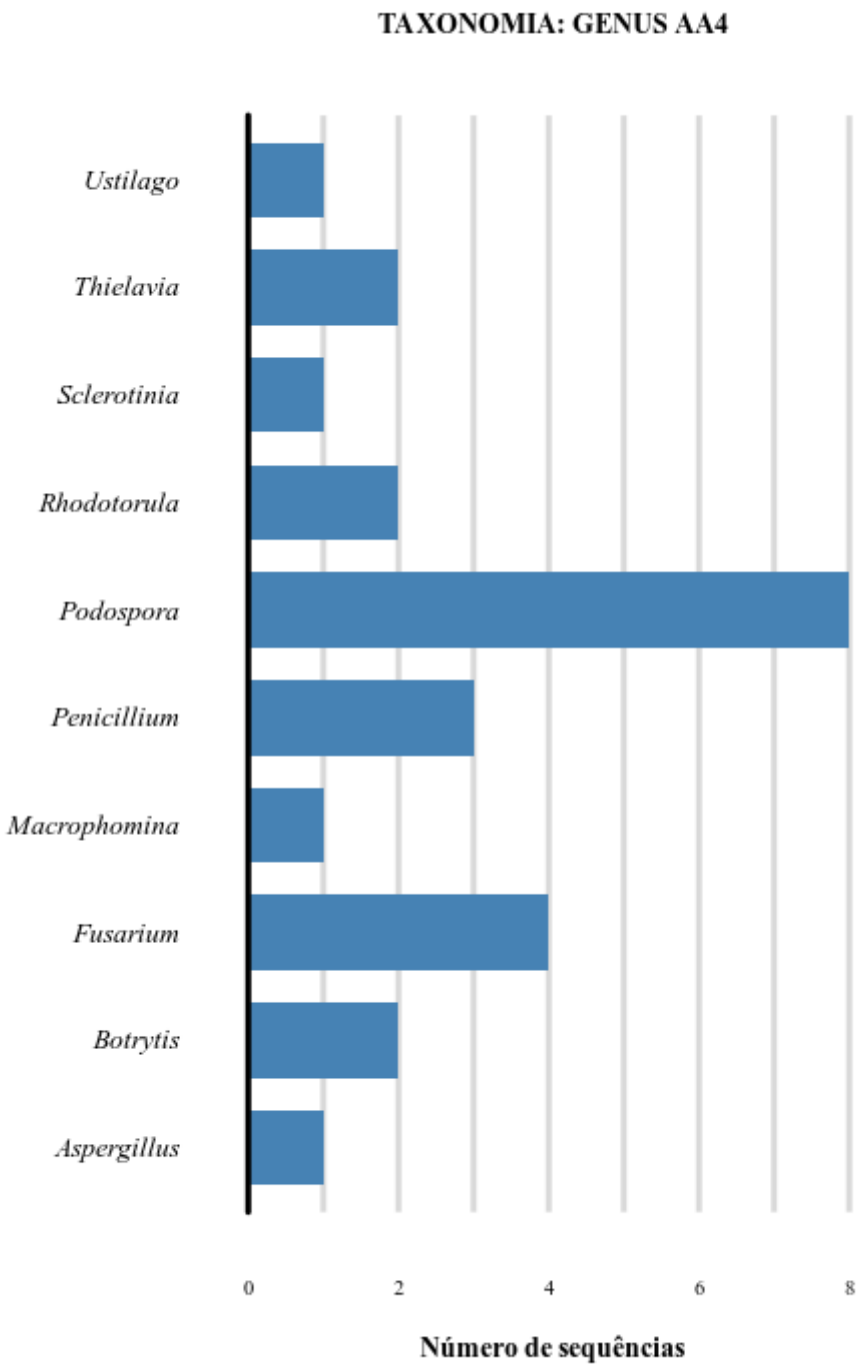

Figura 40 - Gráfico do número de sequências dentro dos gêneros da família AA4. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

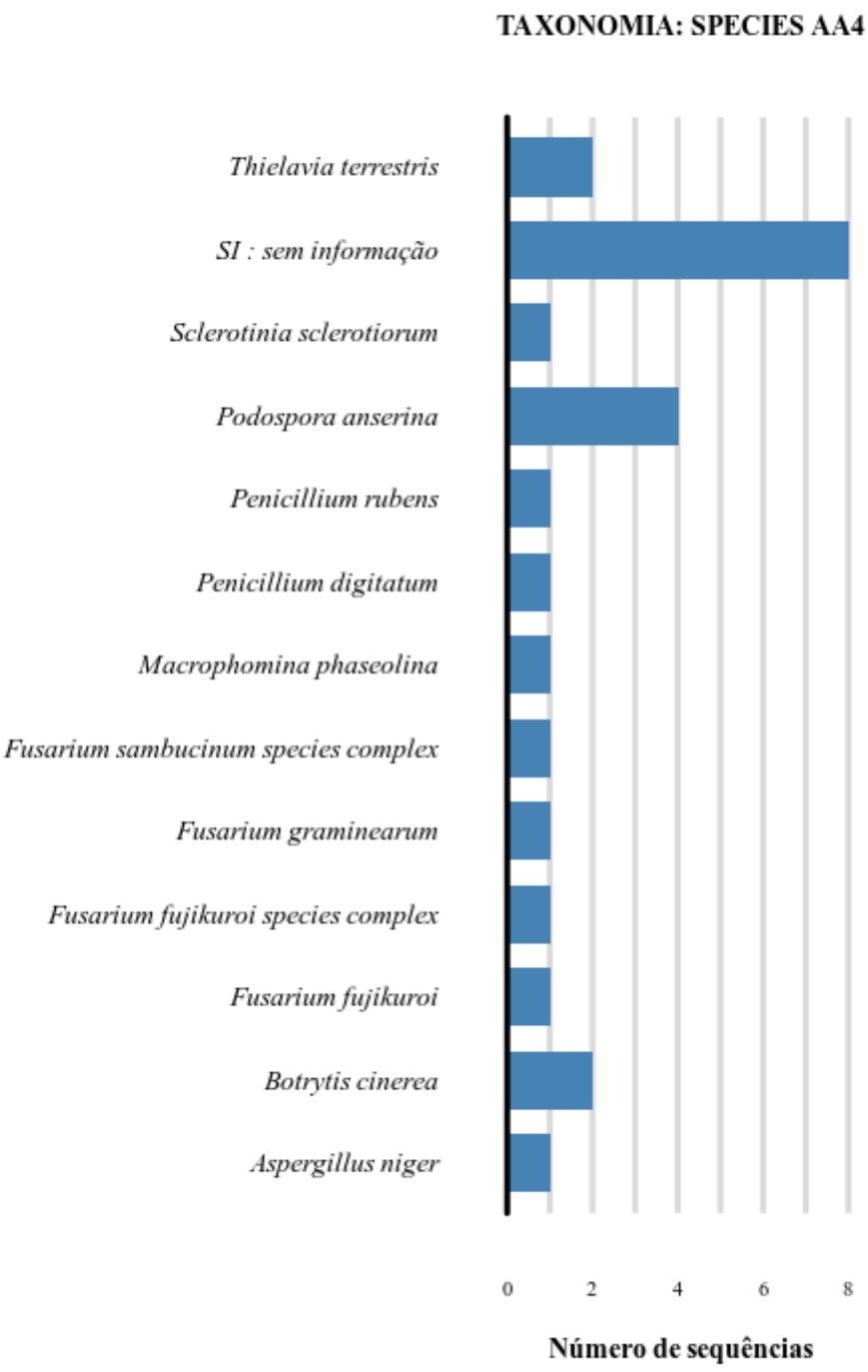

Figura 41 - Gráfico do número de sequências dentro das espécies da família AA4. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

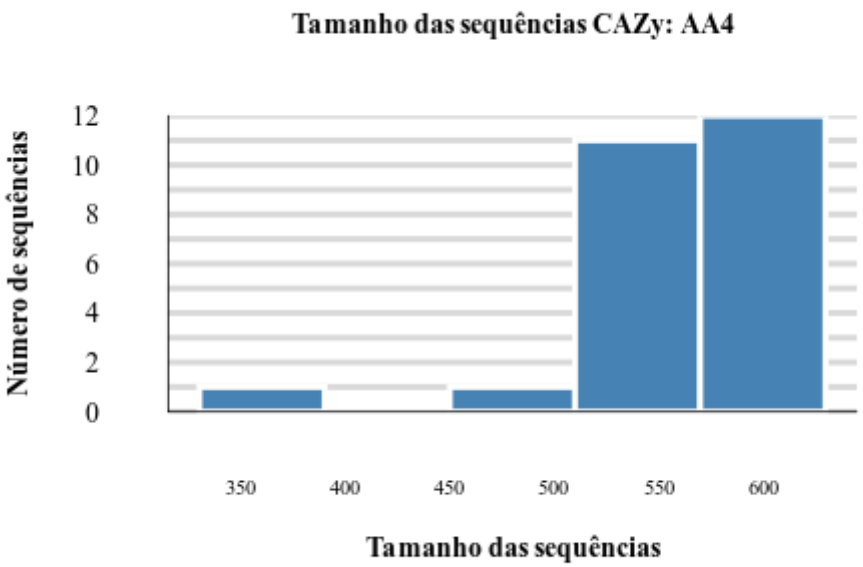

Figura 42 - Gráfico de tamanho de sequência AA4.O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

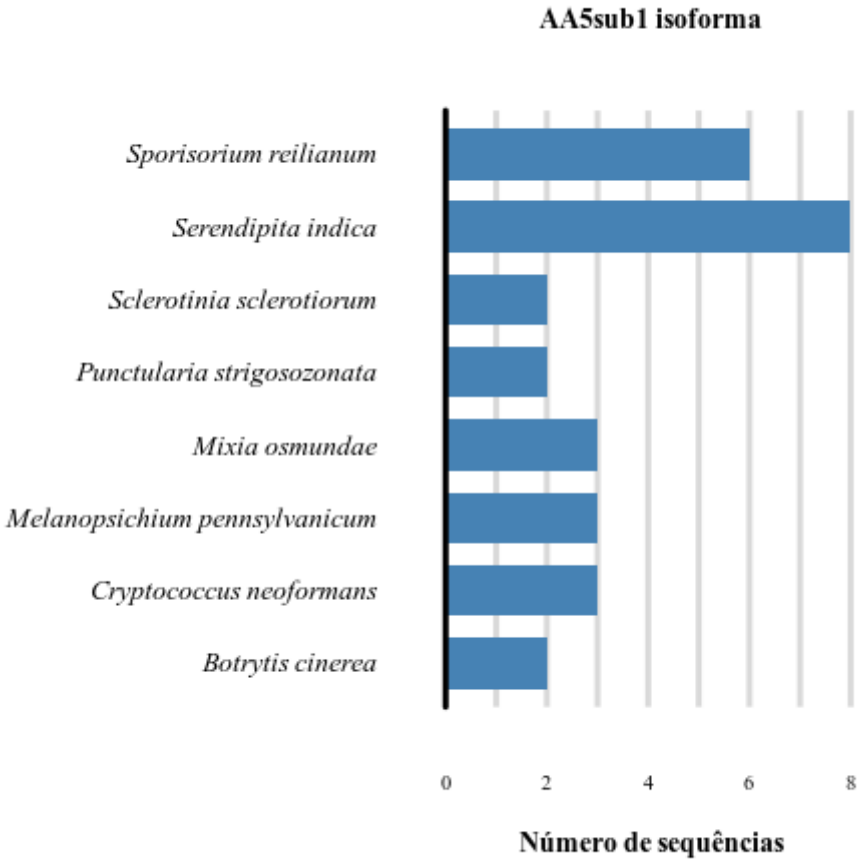

Figura 43 - Gráfico do número de isoformas da subfamília AA5sub1. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

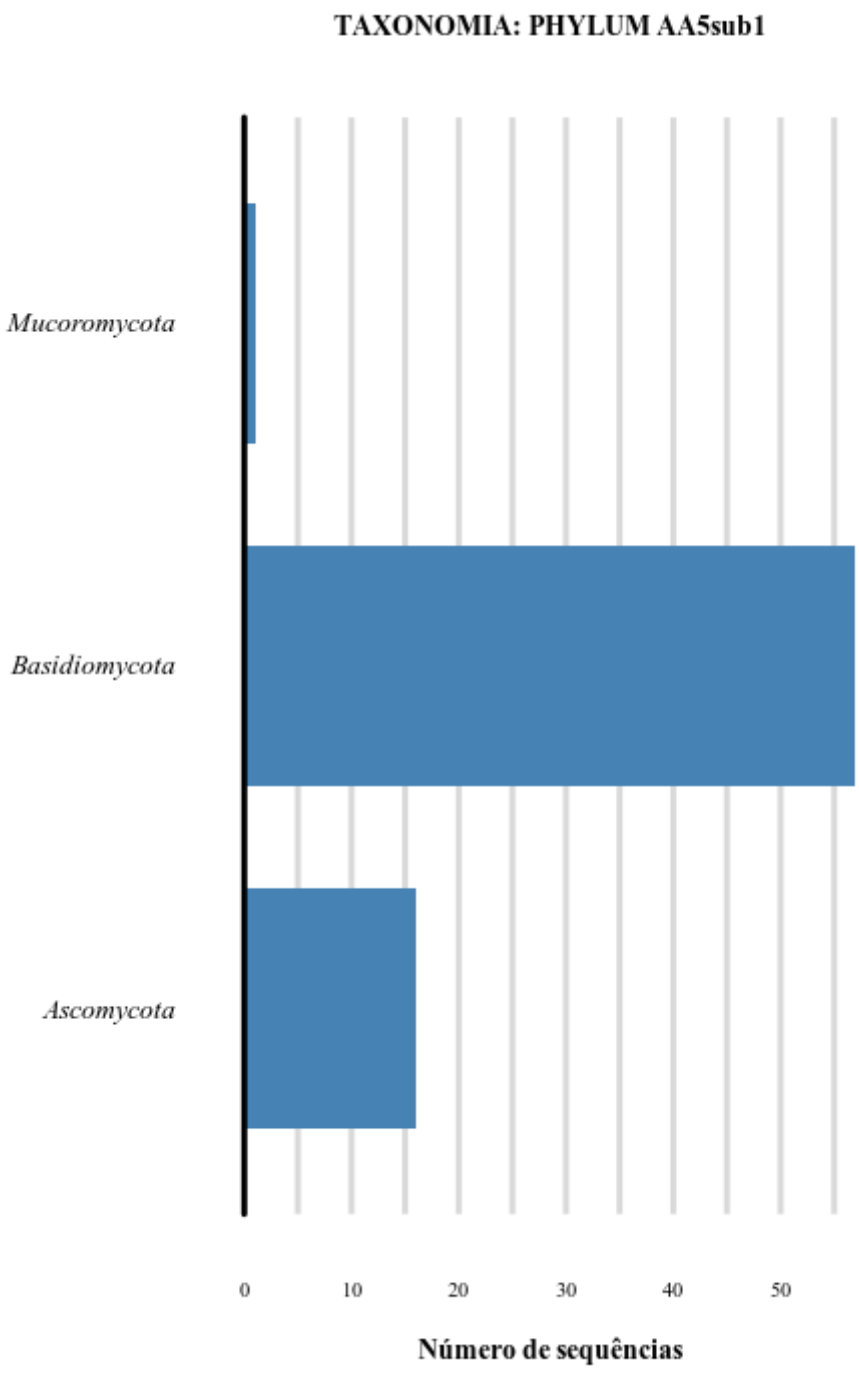

Figura 44 - Gráfico do número de sequências dentro dos filos AA5sub1. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

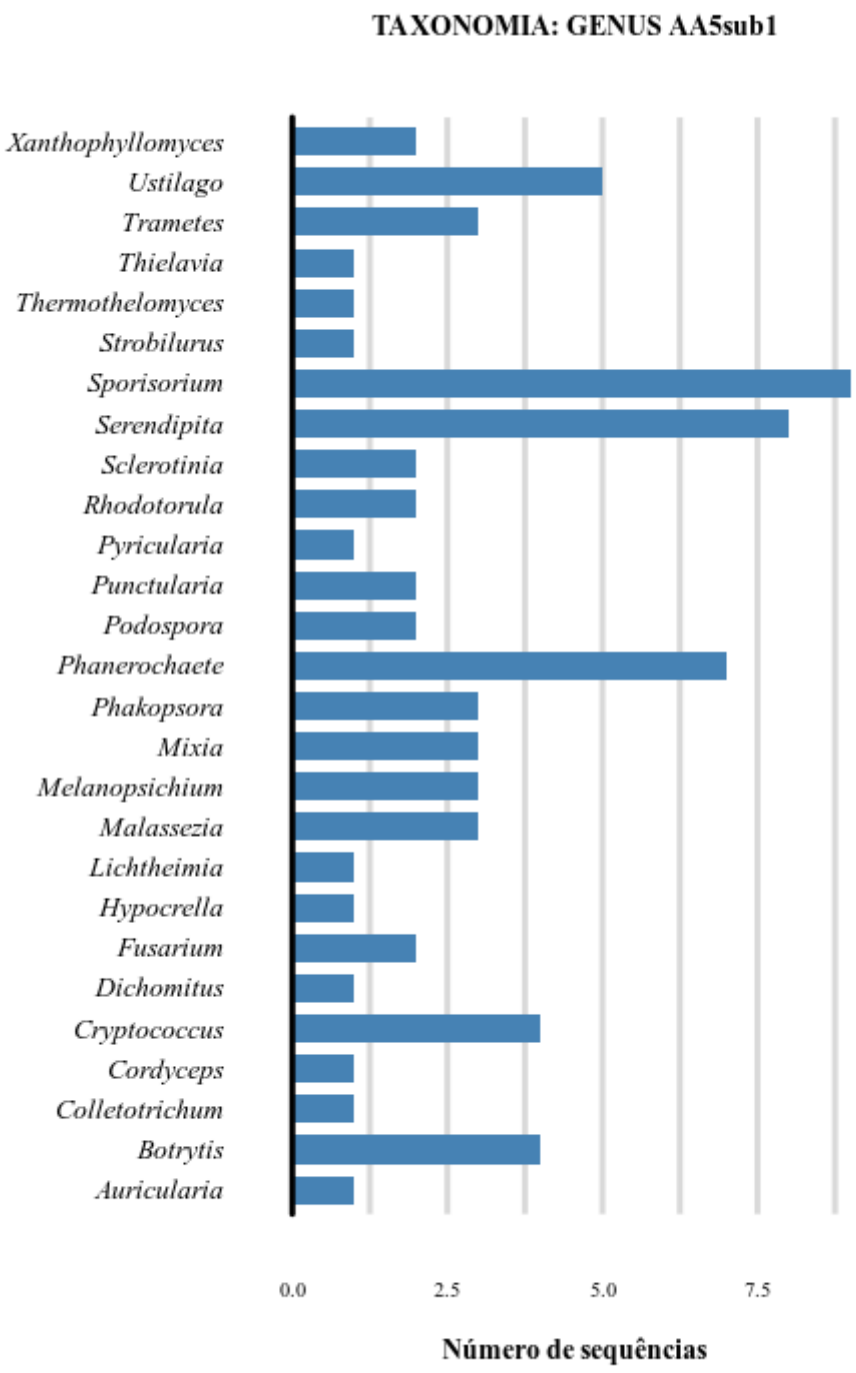

Figura 45 - Gráfico do número de sequências dentro dos gêneros da subfamília AA5sub1. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

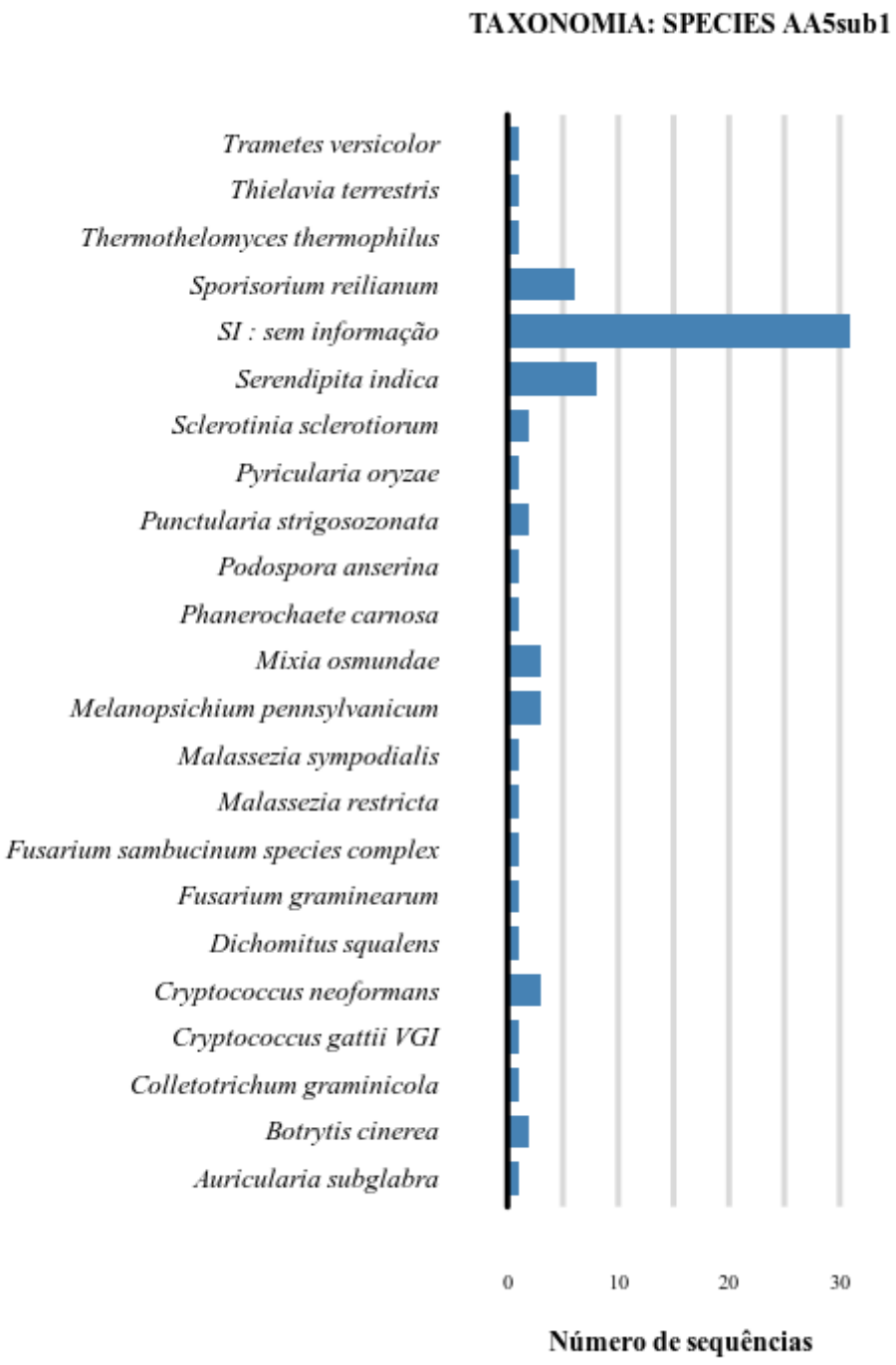

Figura 46 - Gráfico do número de sequências dentro das espécies da subfamília AA5sub1. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

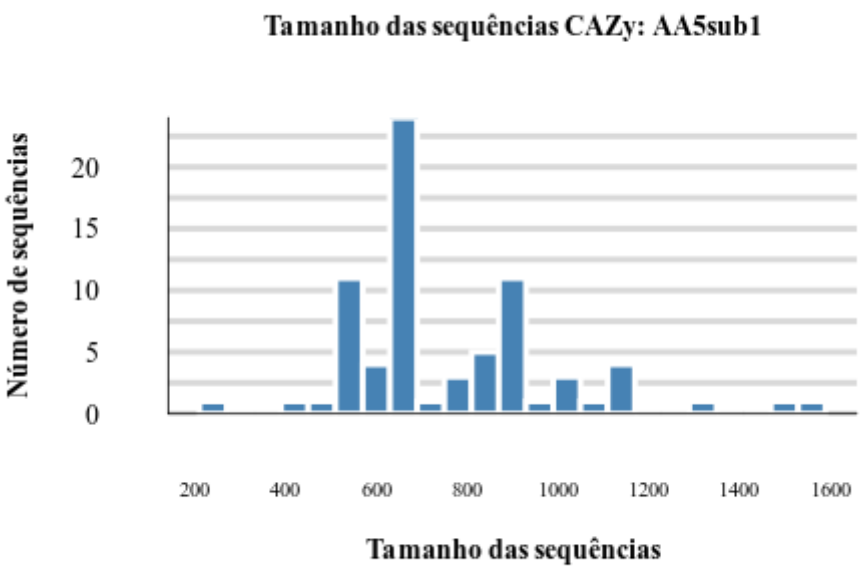

Figura 47 - Gráfico de tamanho de sequências da subfamília AA5sub1. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

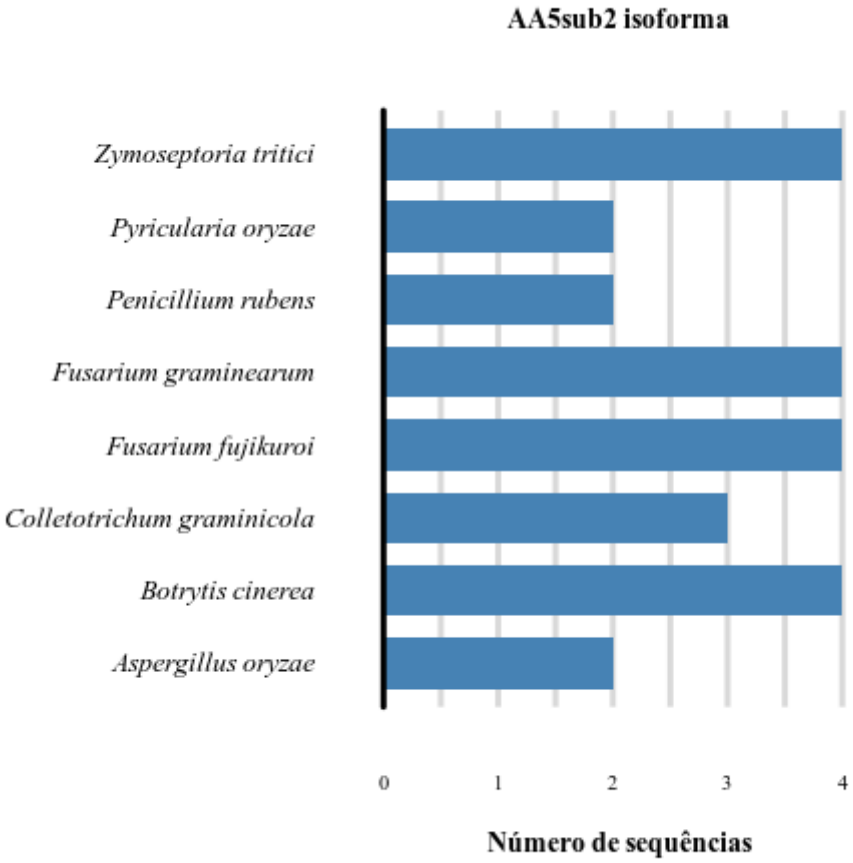

Figura 48 - Gráfico do número de isoformas da subfamília AA5sub2. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

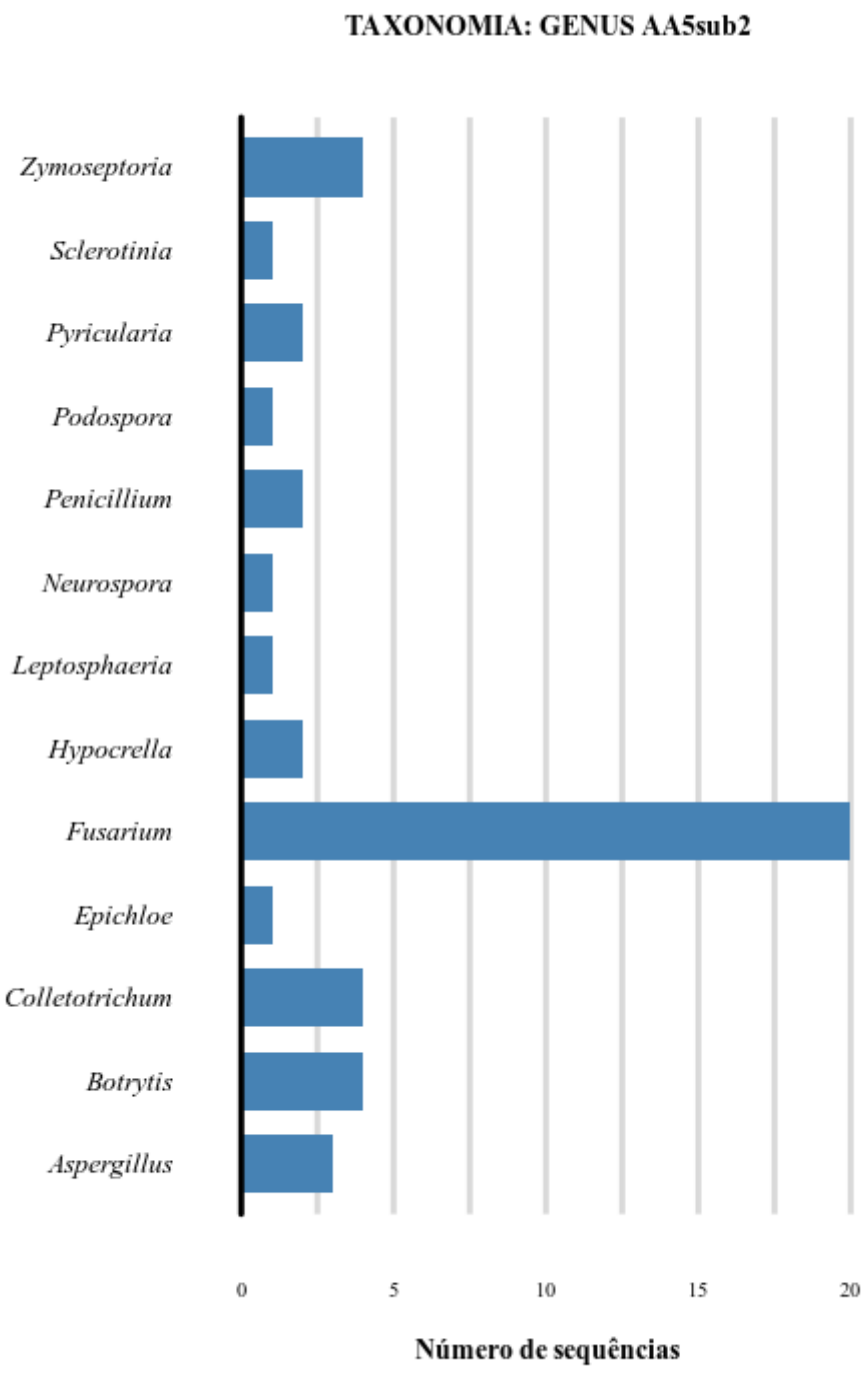

Figura 49 - Gráfico do número de sequências dentro dos gêneros da subfamília AA5sub2. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

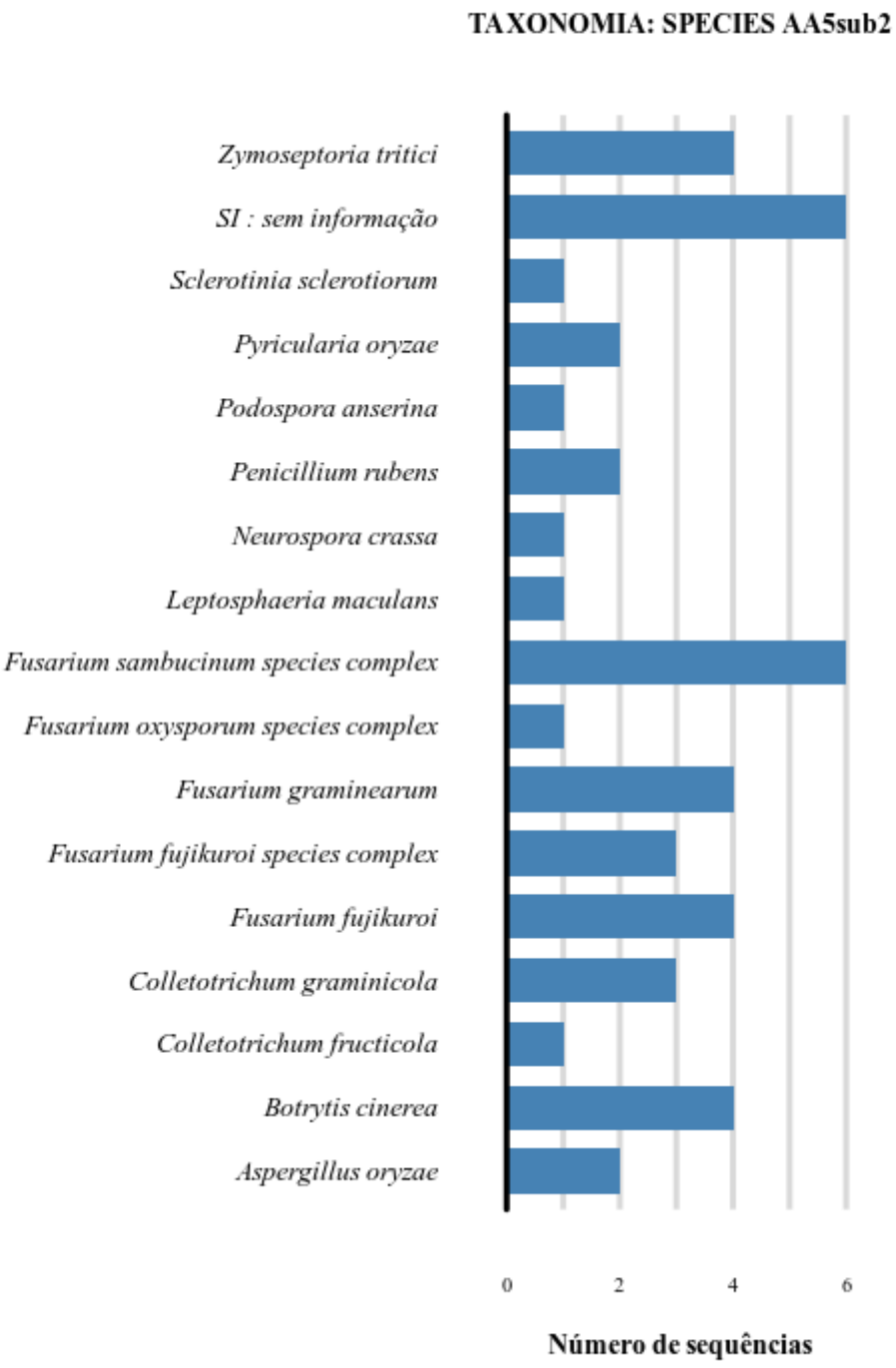

Figura 50 - Gráfico do número de sequências dentro das espécies da subfamília AA5sub2. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

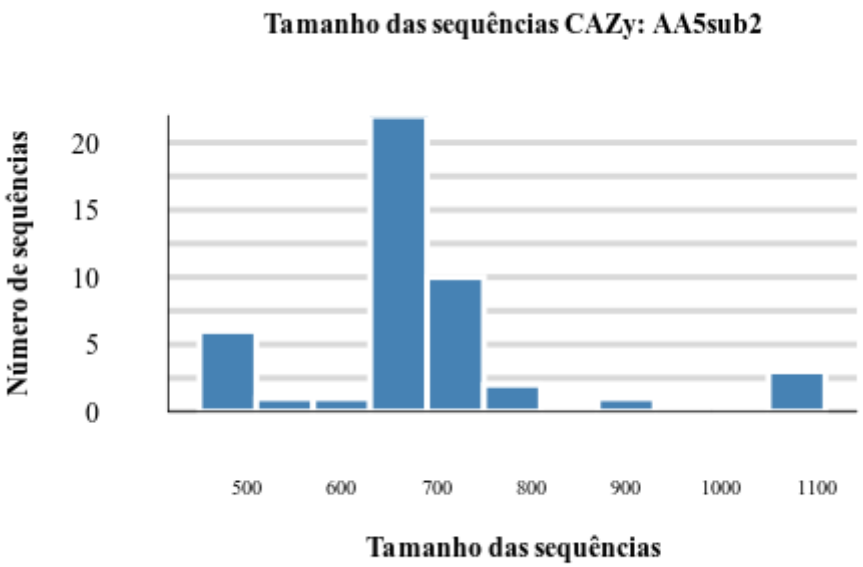

Figura 51 - Gráfico de tamanho de sequência da subfamília AA5sub2. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

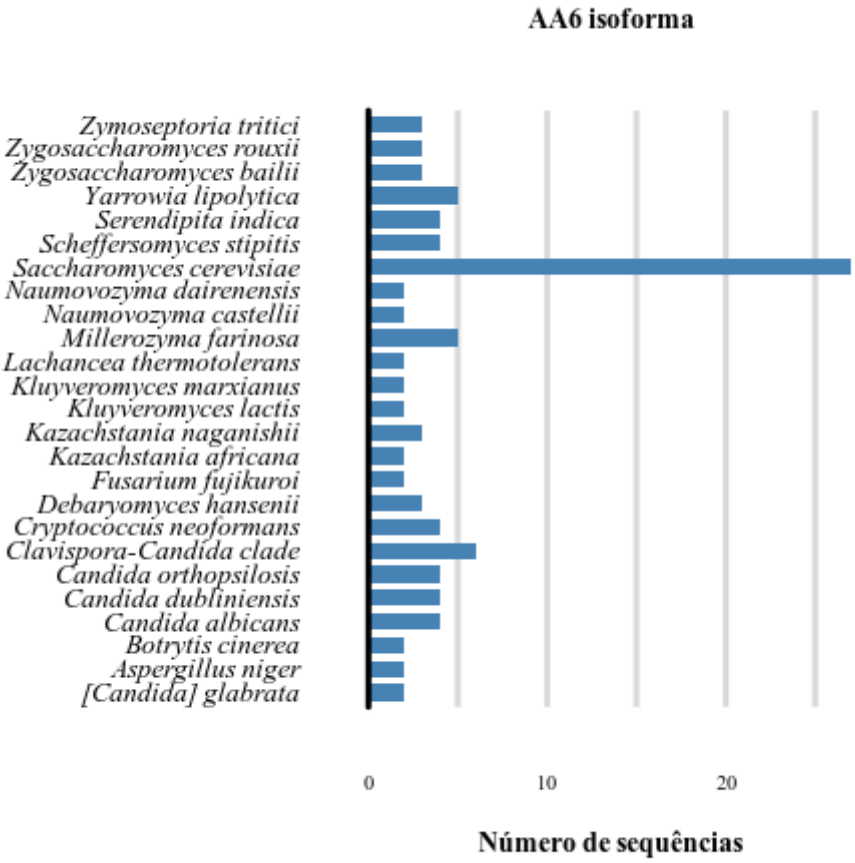

Figura 52 - Gráfico do número de isoformas da família AA6. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

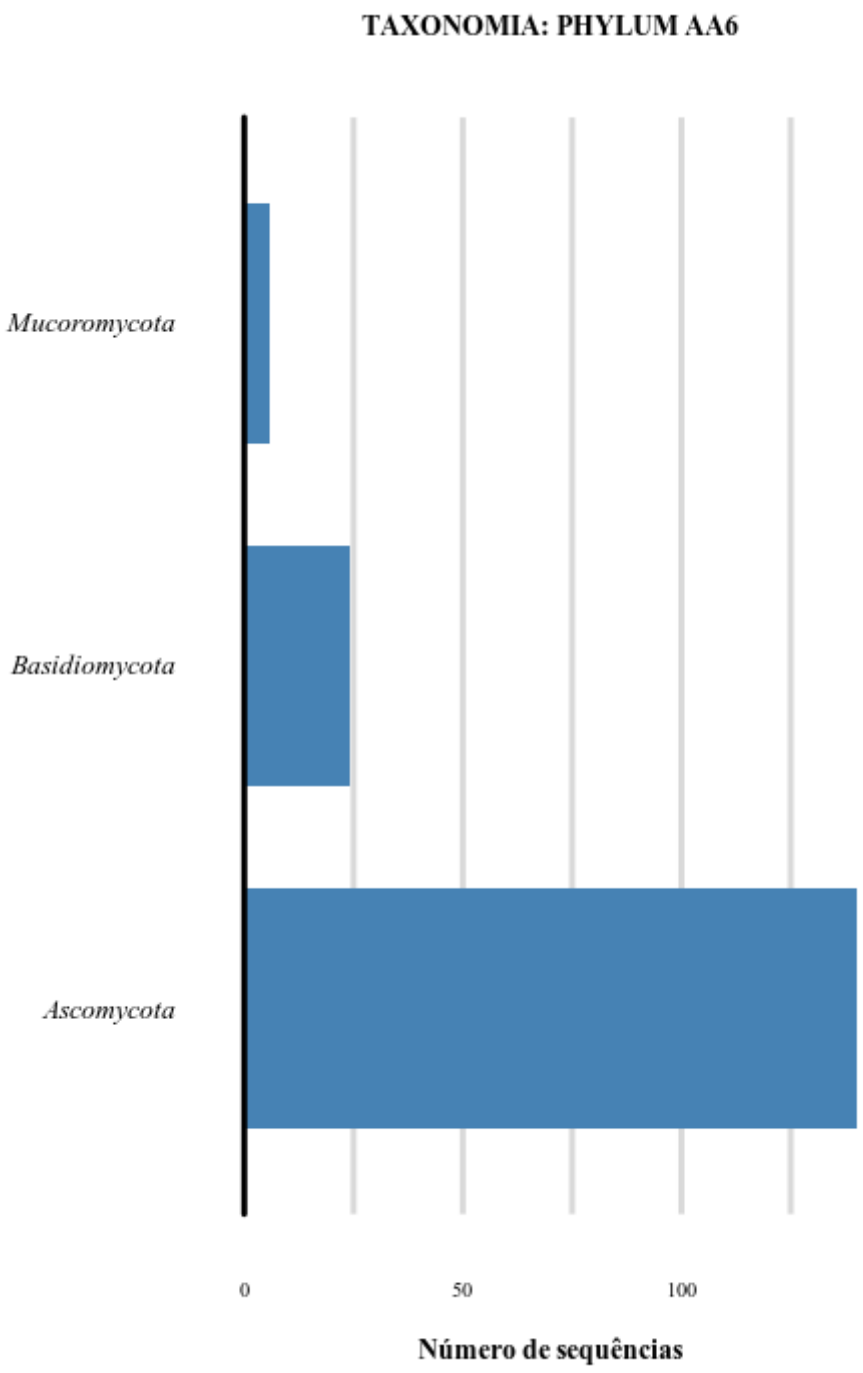

Figura 53 - Gráfico do número de sequências dentro dos filos da família AA6. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

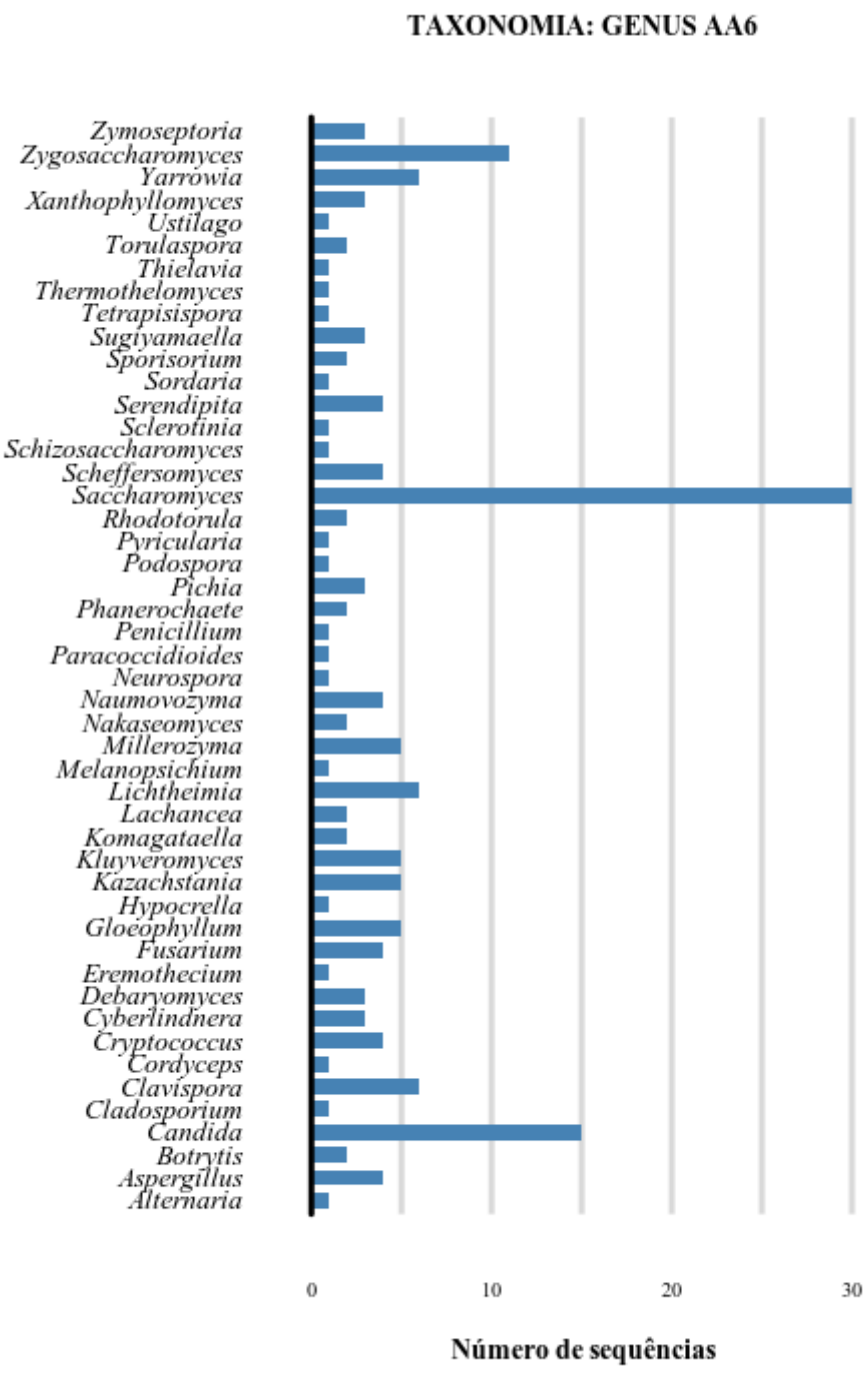

Figura 54 - Gráfico do número de sequências dentro dos gêneros da família AA6. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

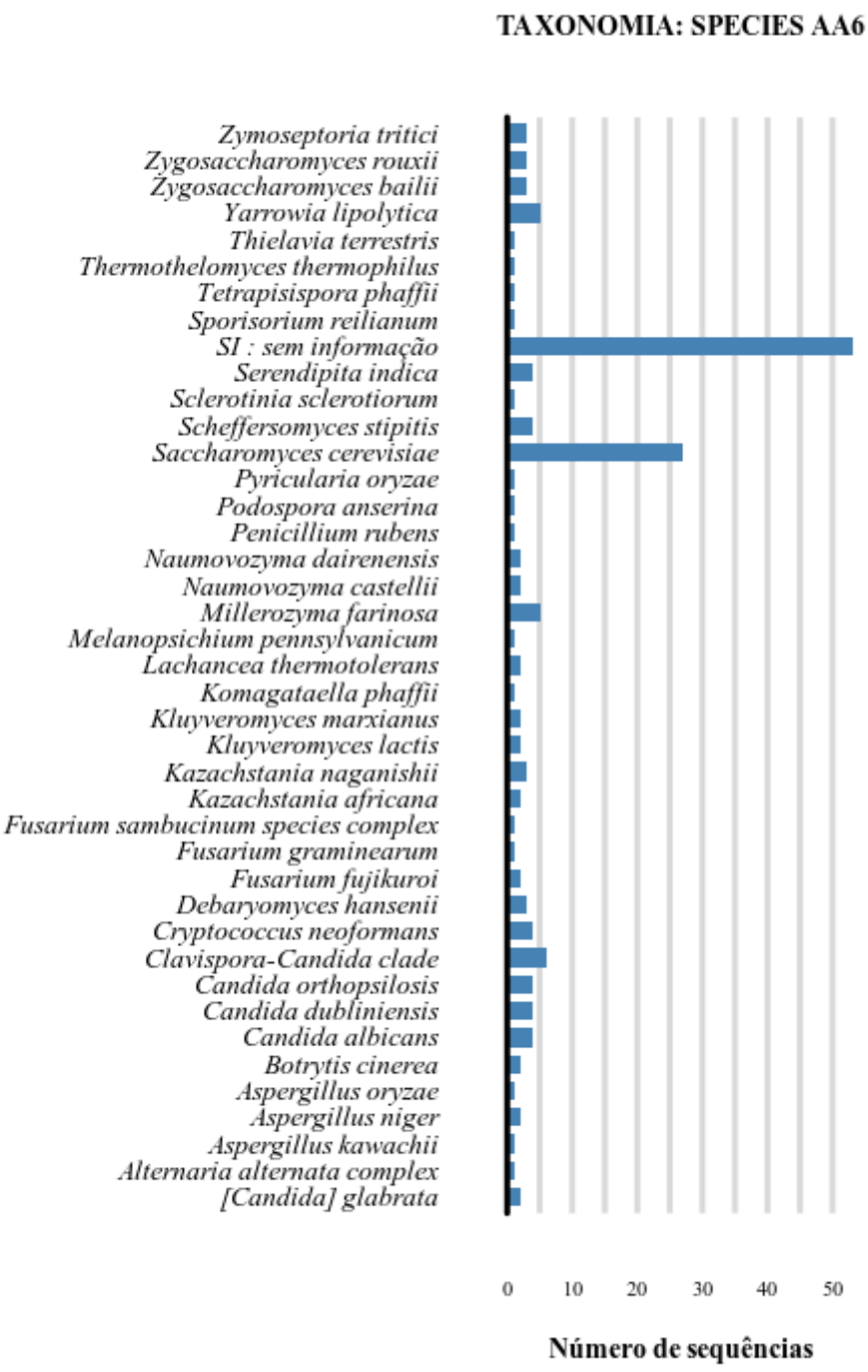

Figura 55 - Gráfico do número de sequências dentro das espécies da família AA6. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

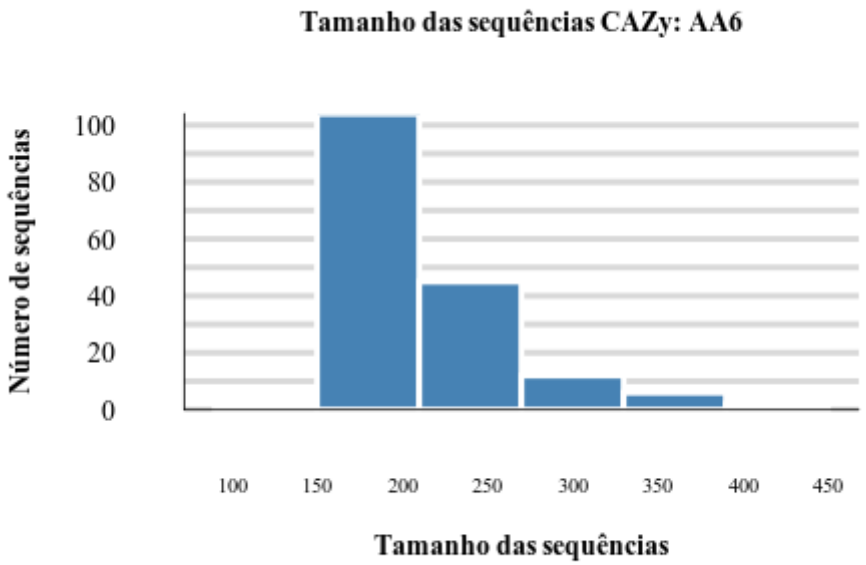

Figura 56 - Gráfico de tamanho de sequências da família AA6. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

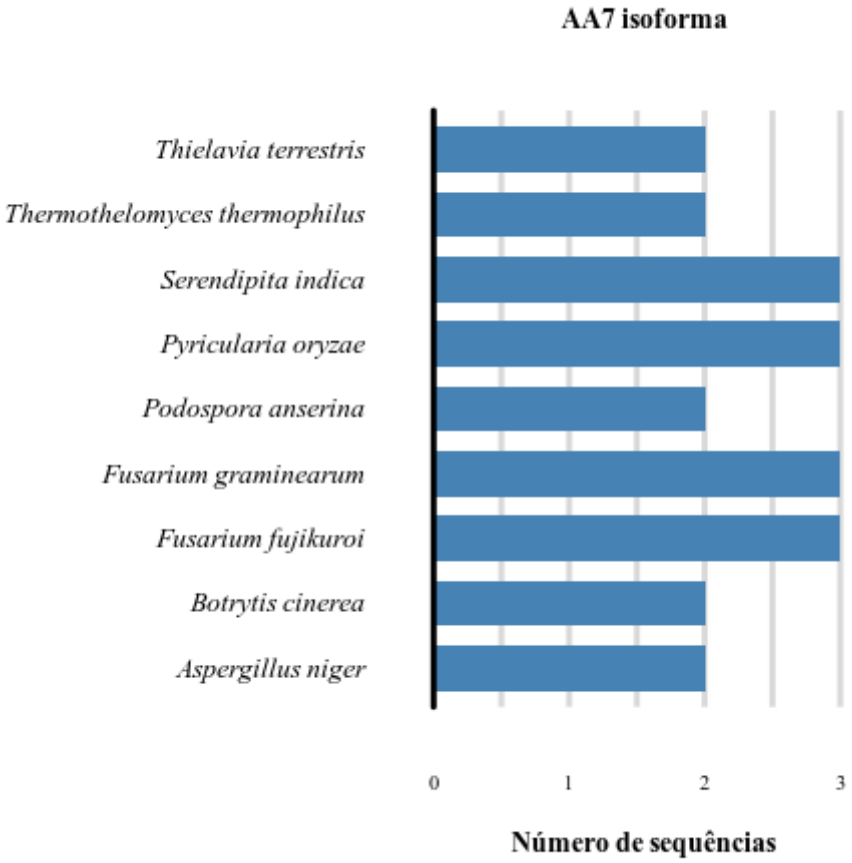

Figura 57 - Gráfico do número de isoformas da família AA7. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

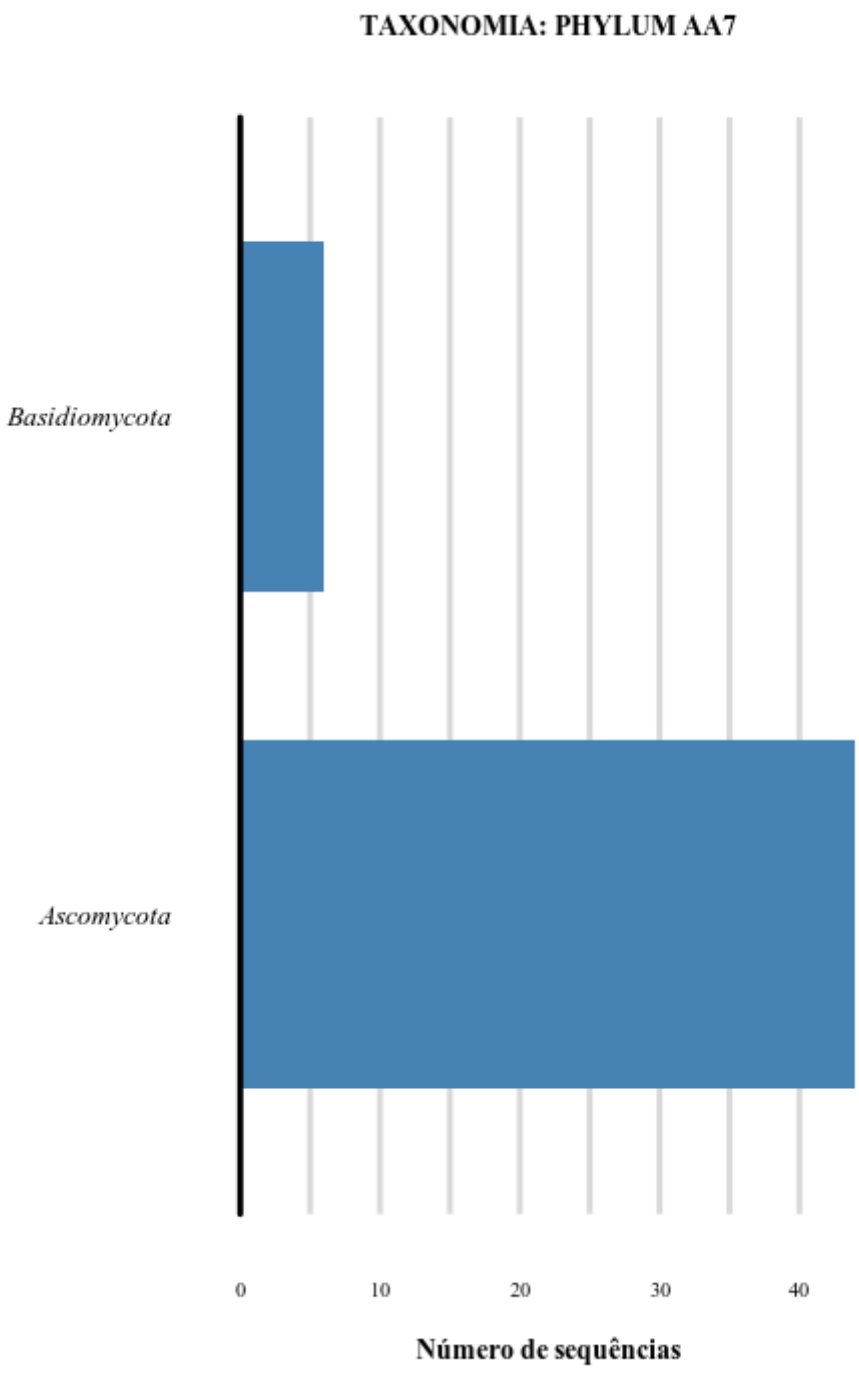

Figura 58 - Gráfico do número de sequências dentro dos filos da família AA7. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

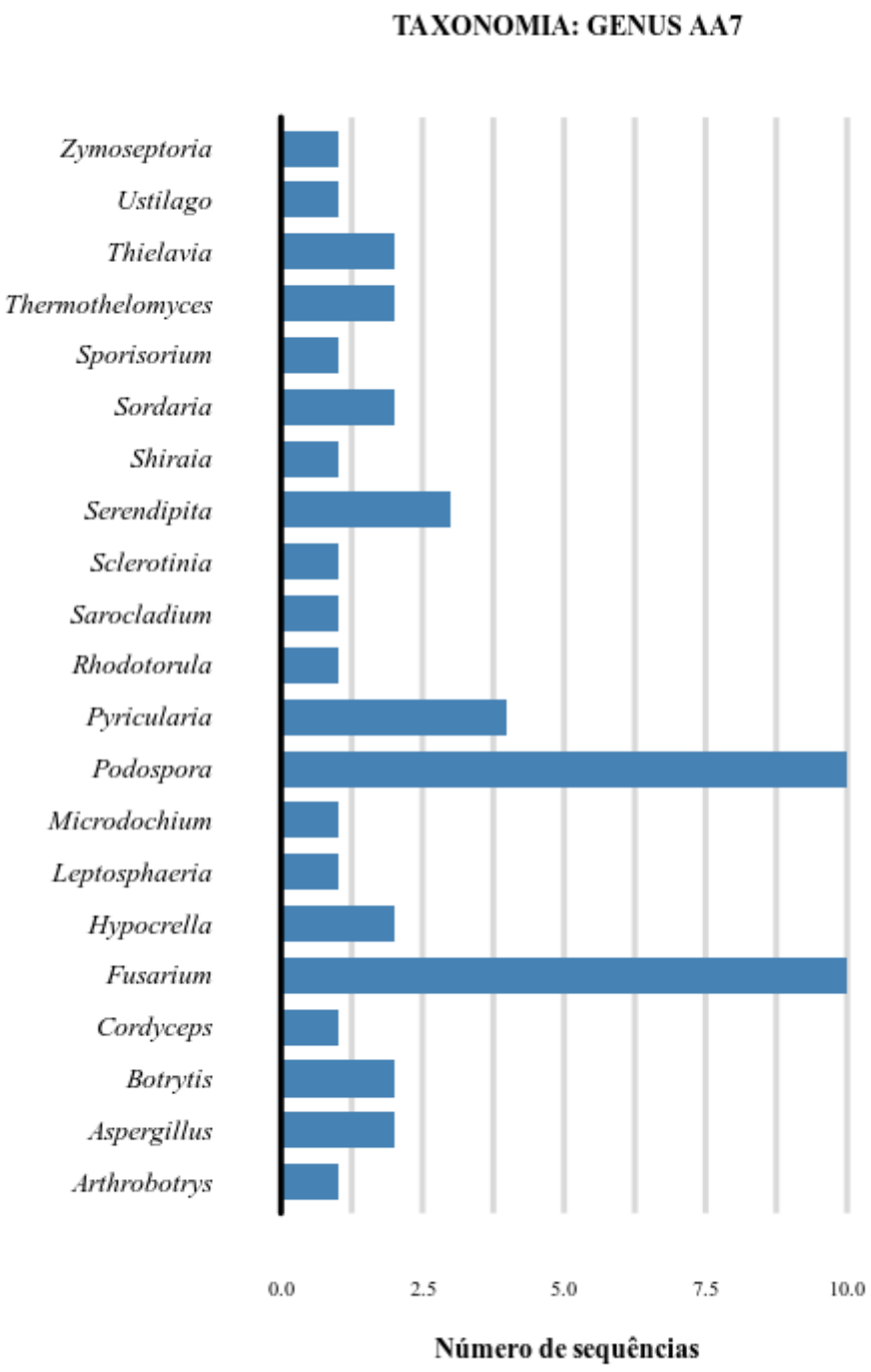

Figura 59 - Gráfico do número de sequências dentro dos gêneros da família AA7. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

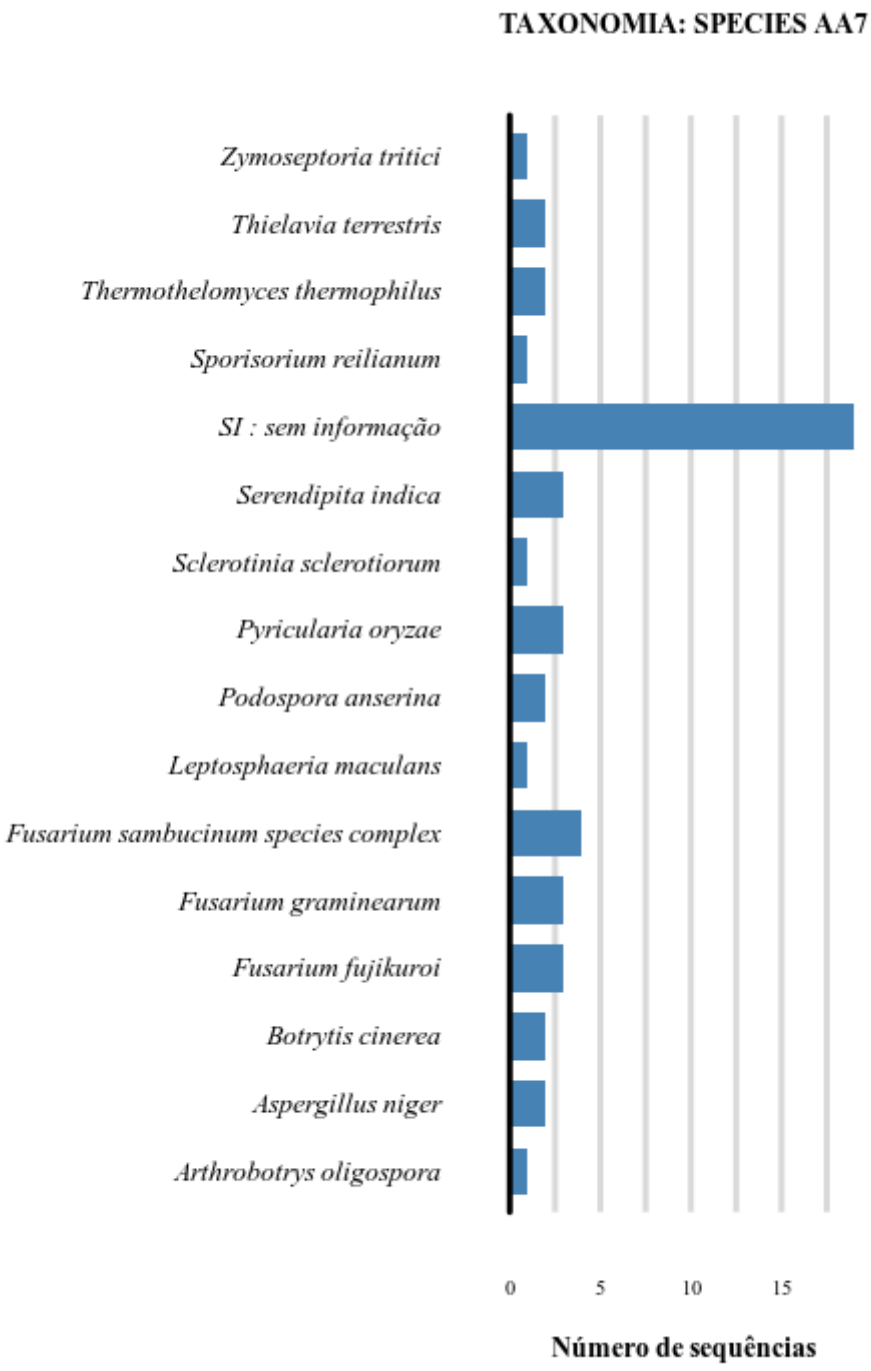

Figura 60 - Gráfico do número de sequências dentro das espécies AA7. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

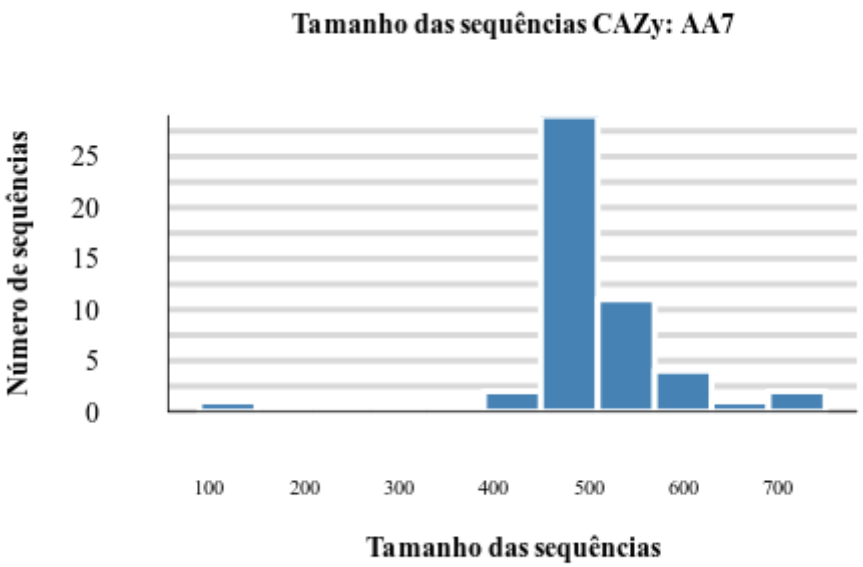

Figura 61 - Gráfico de tamanho de sequência AA7. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

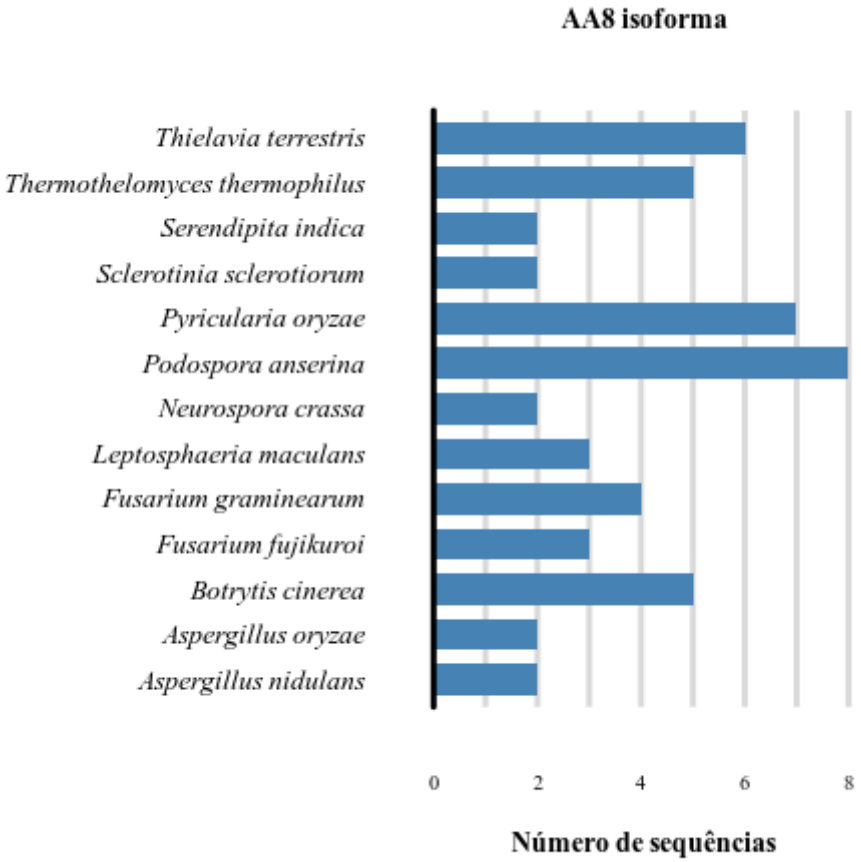

Figura 62 - Gráfico do número de isoformas da família AA8.Gráfico O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

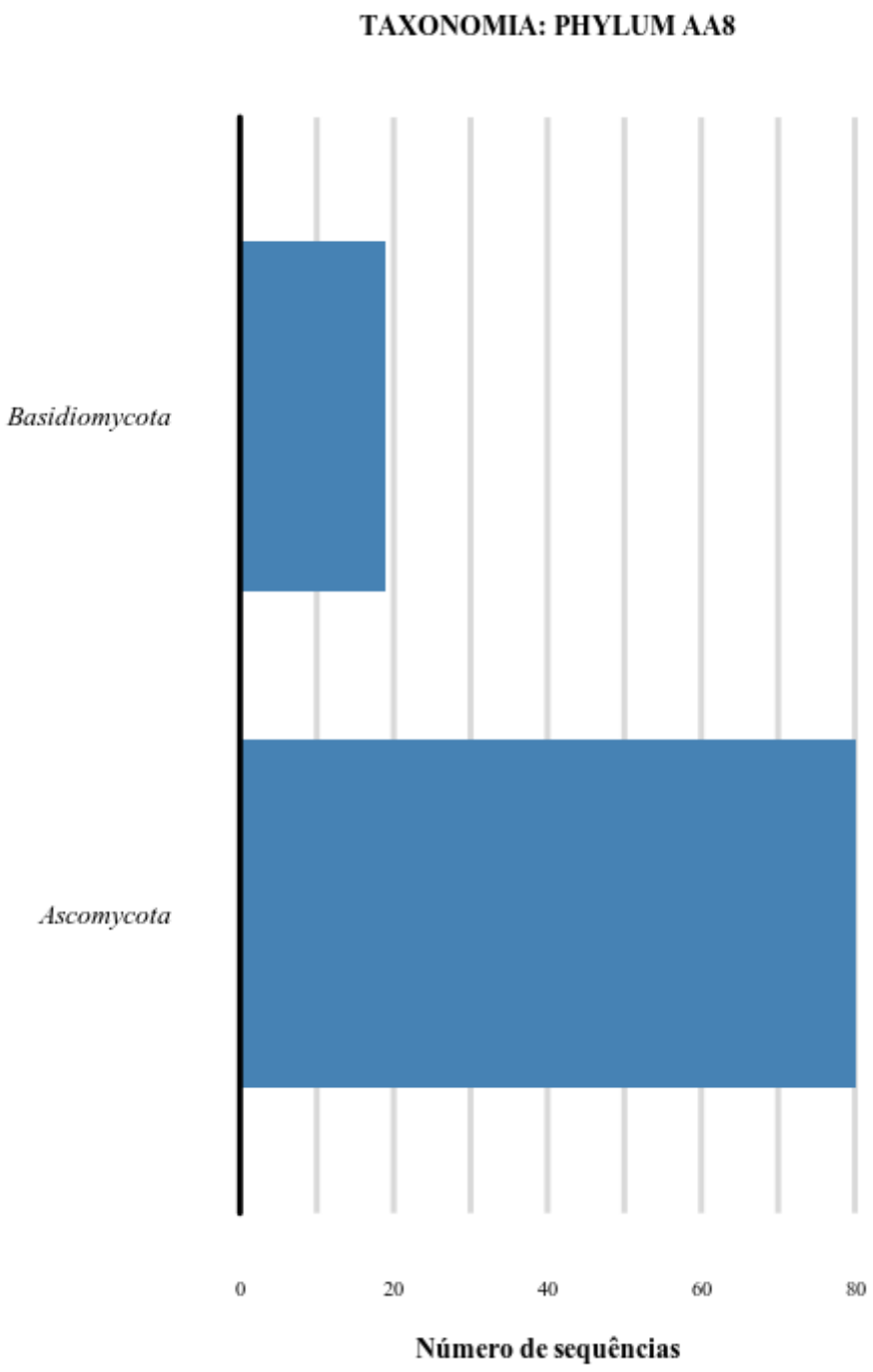

Figura 63 - Gráfico do número de sequências dentro dos filos da família AA8. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

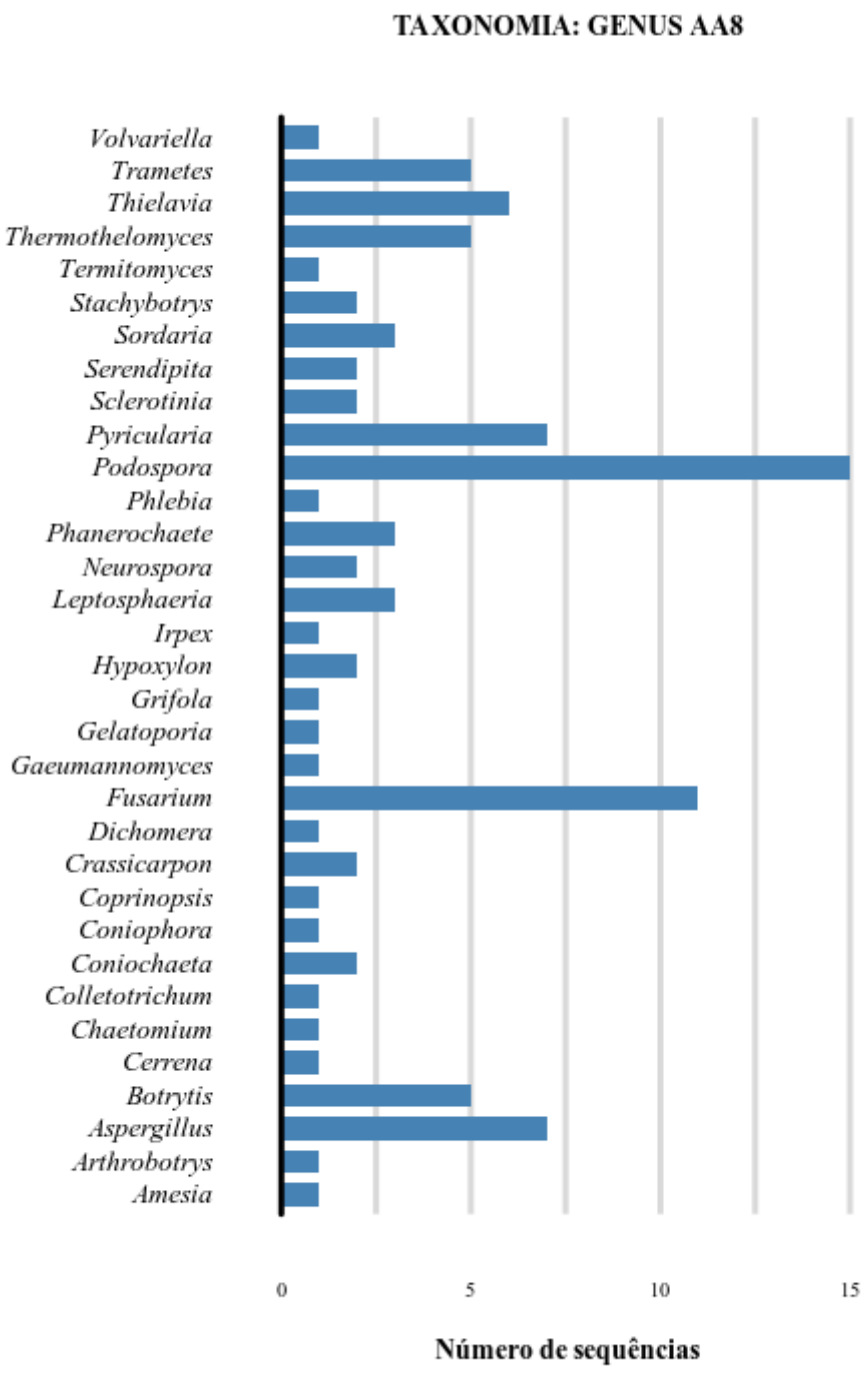

Figura 64 - Gráfico do número de sequências dentro dos gêneros da família AA8. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

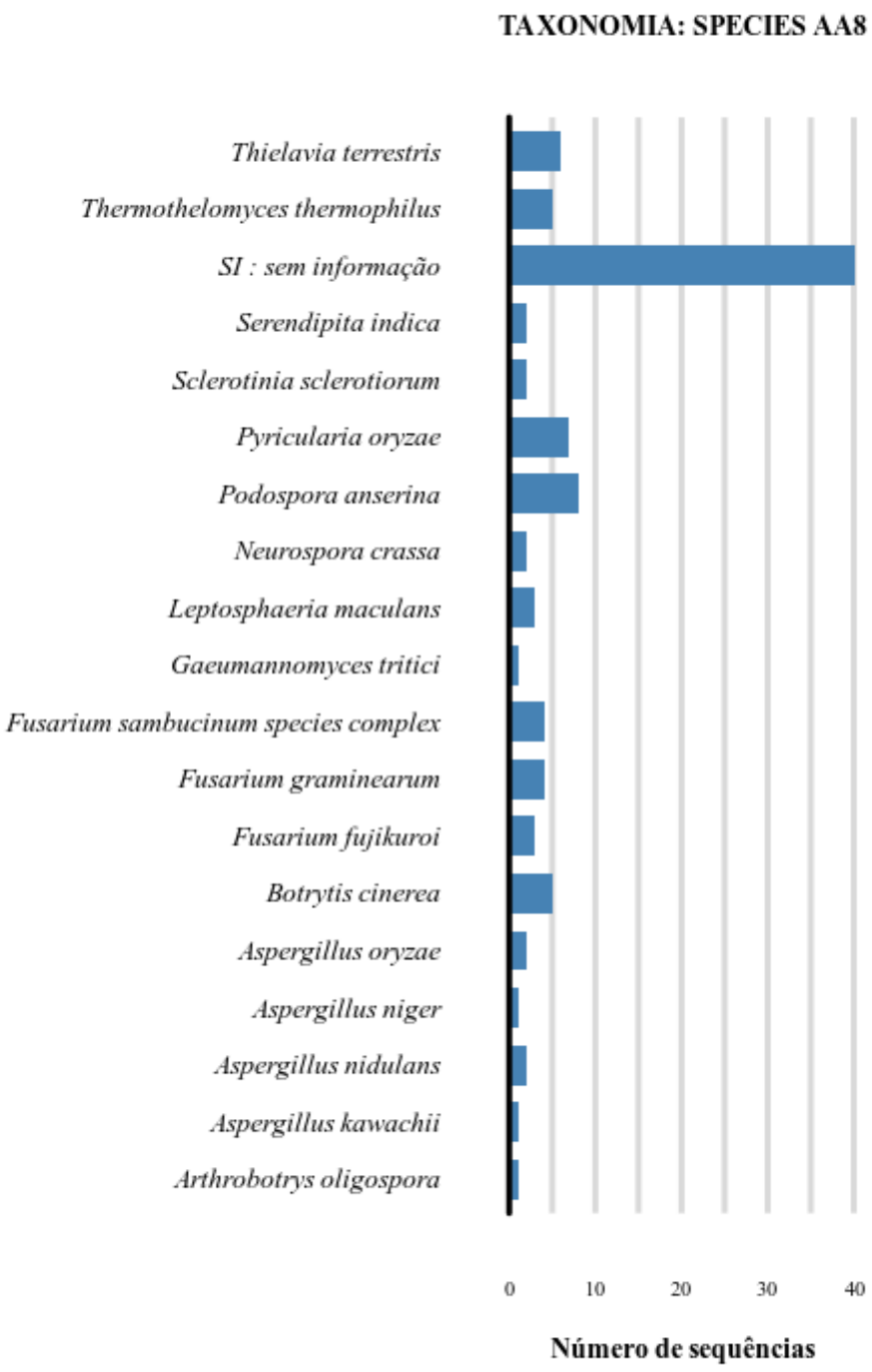

Figura 65 - Gráfico do número de sequências dentro das espécies da família AA8. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

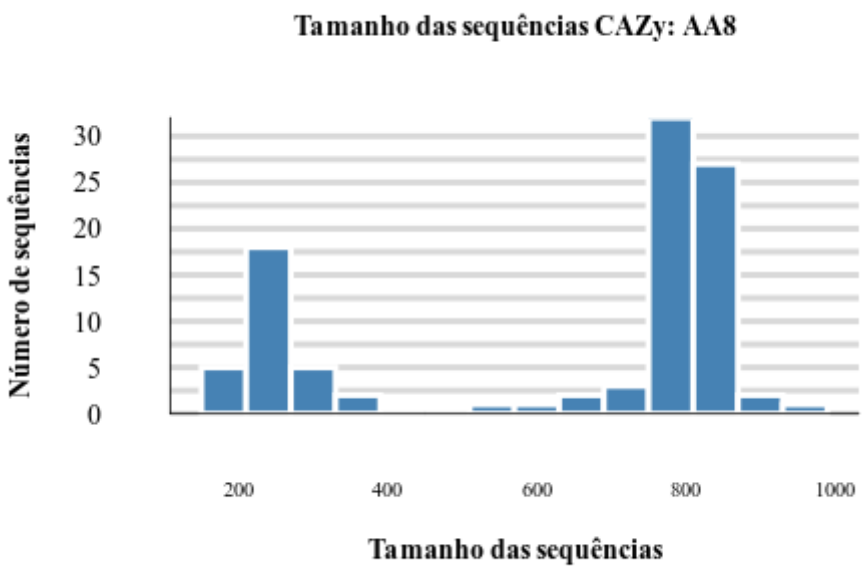

Figura 66 - Gráfico de tamanhos de sequência da família AA8. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

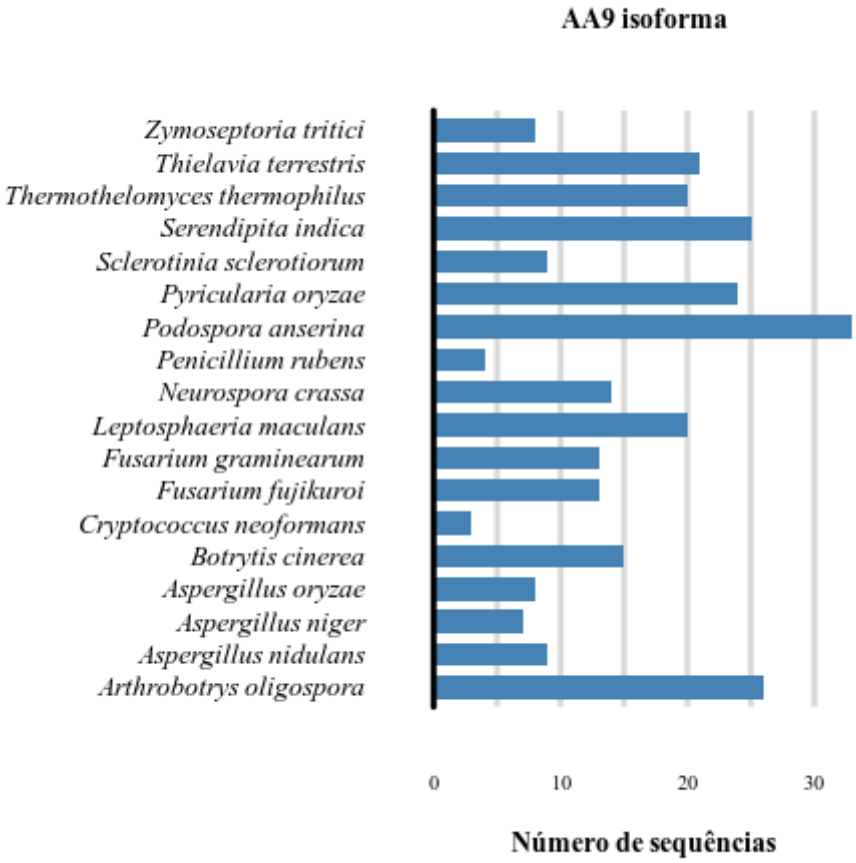

Figura 67 - Gráfico do número de isoformas da família AA9. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

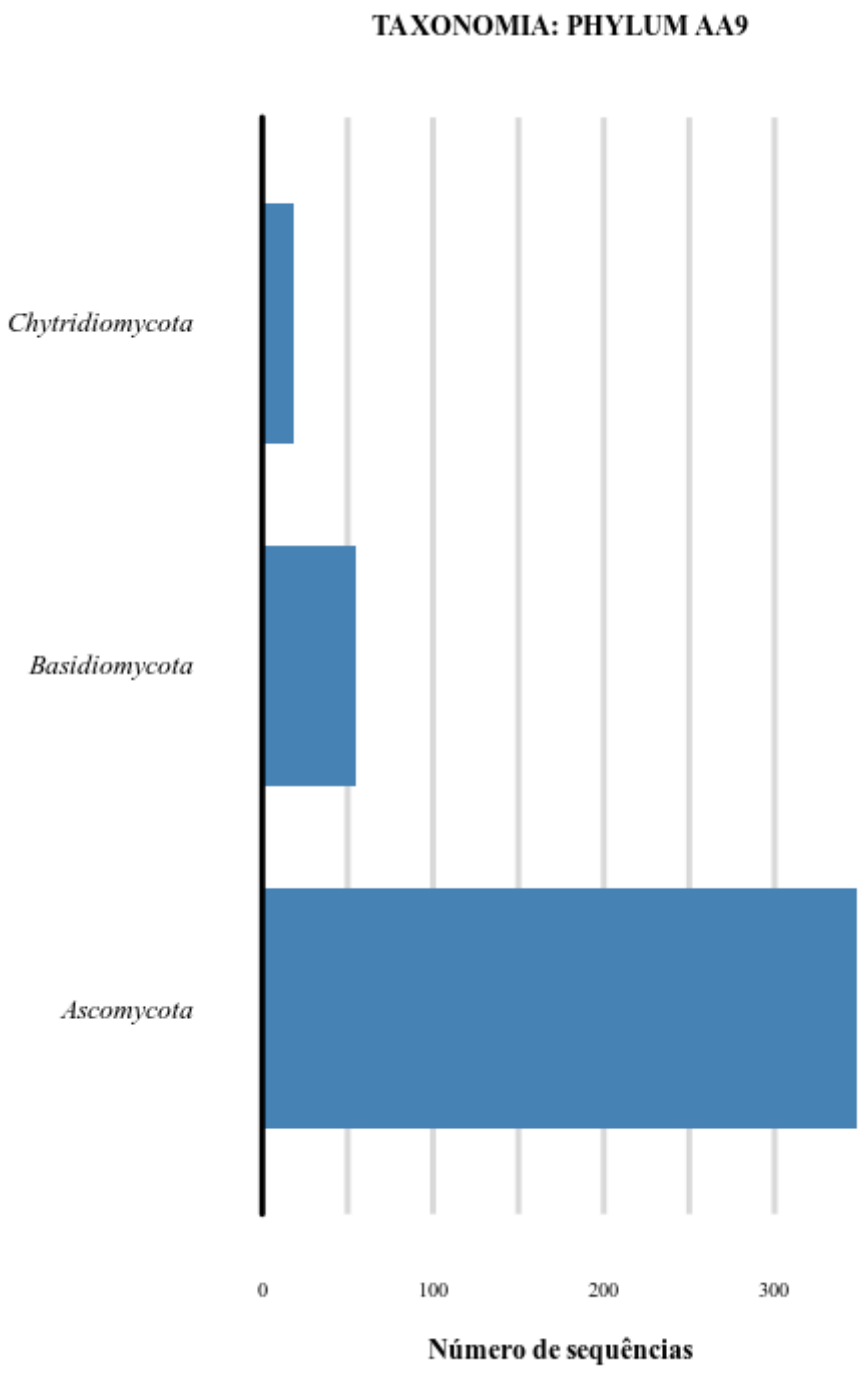

Figura 68 - Gráfico do número de sequências dentro dos filos AA9. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

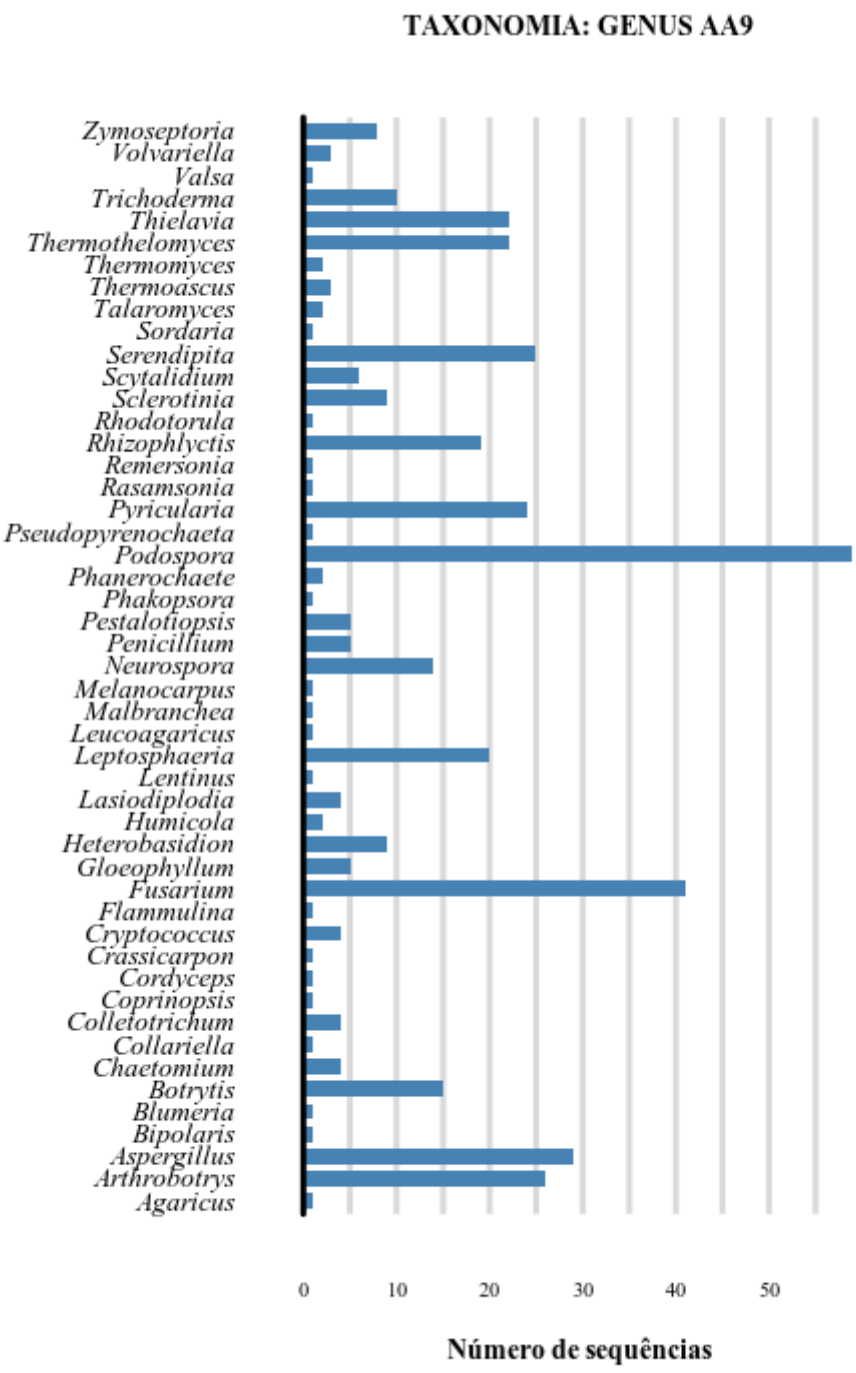

Figura 69 - Gráfico do número de sequências dentro dos gêneros da família AA9. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

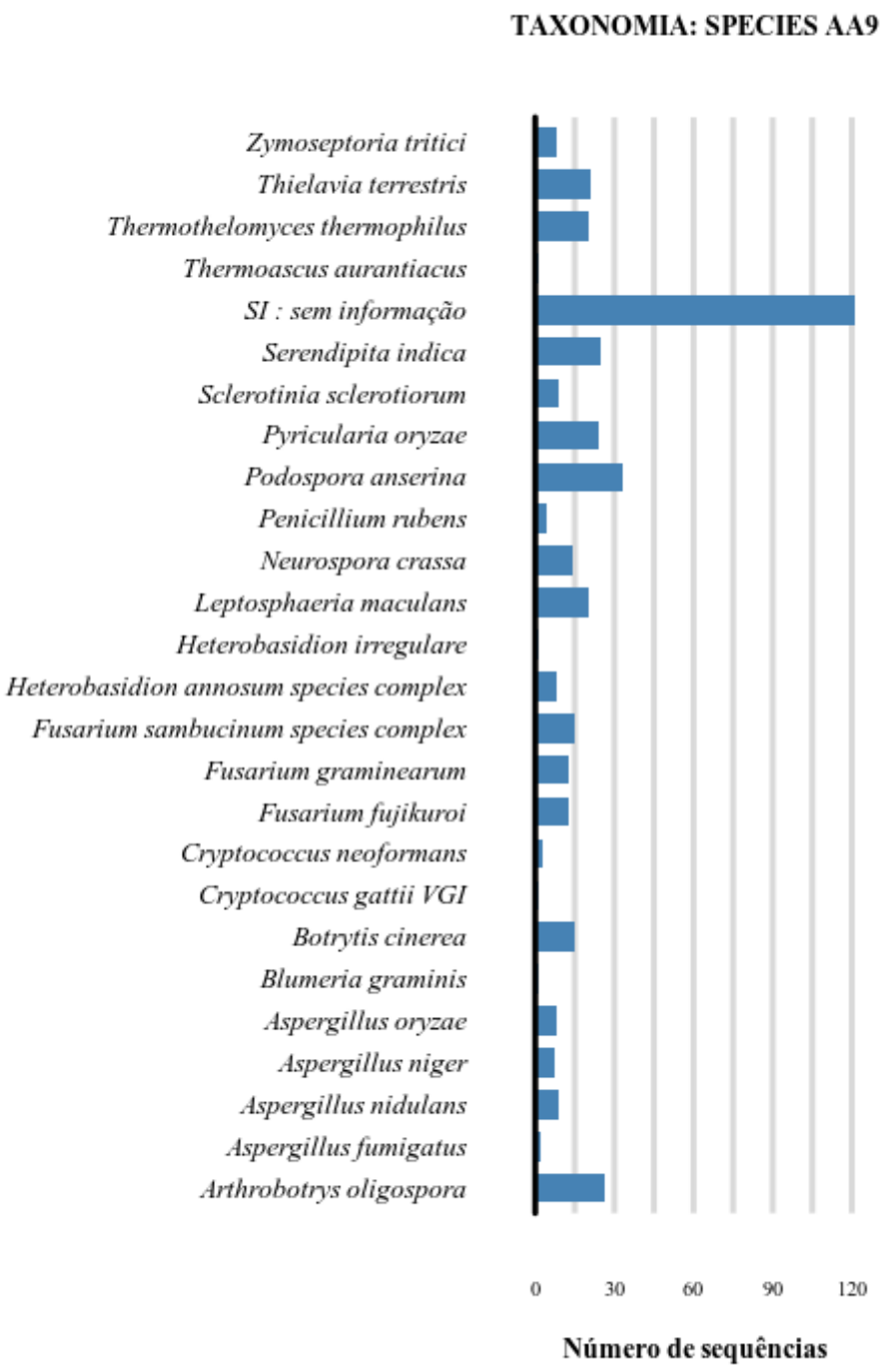

Figura 70 - Gráfico do número de sequências dentro das espécies da família AA9. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

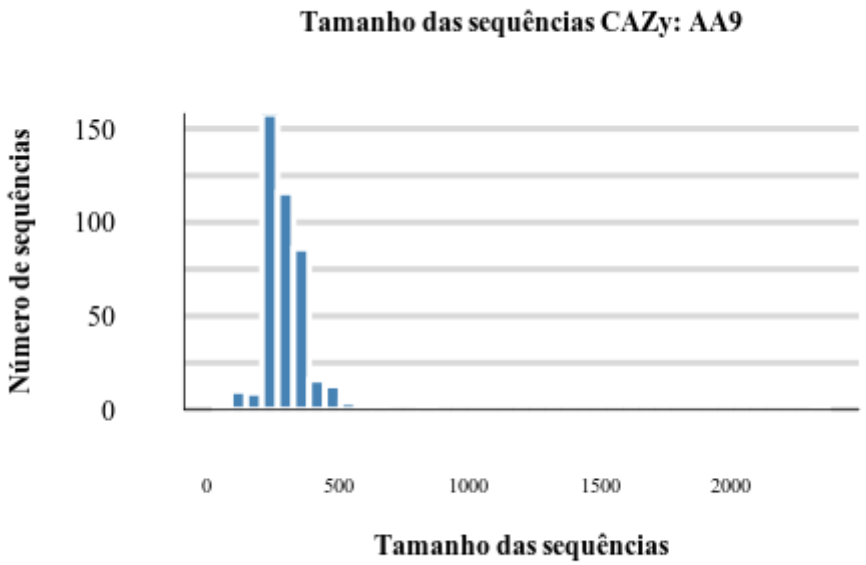

Figura 71 - Gráfico de tamanho de sequências da família AA9. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

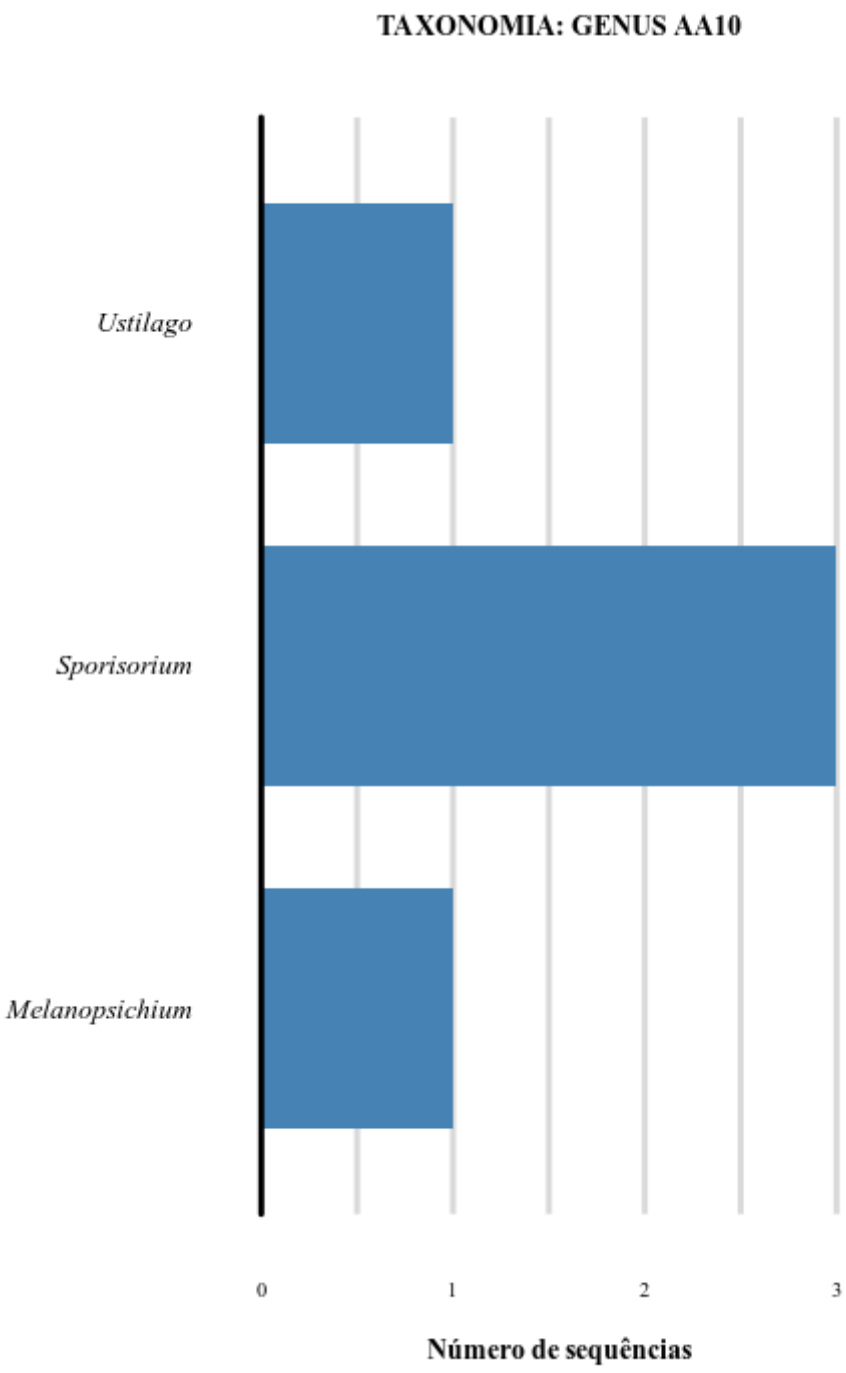

Figura 72 - Gráfico do número de sequências dentro dos gêneros da família AA10. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

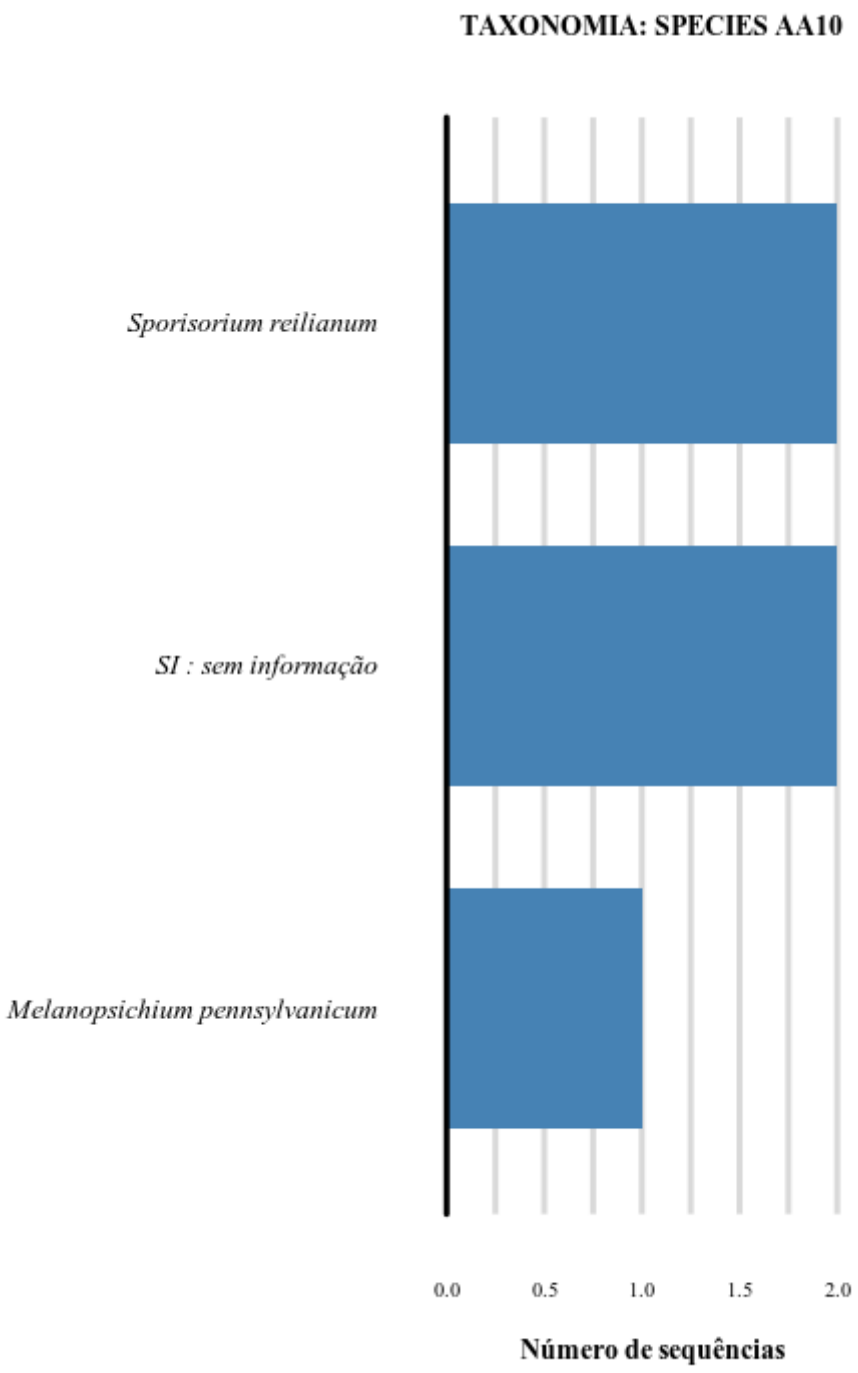

Figura 73 - Gráfico do número de sequências dentro das espécies da família AA10. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

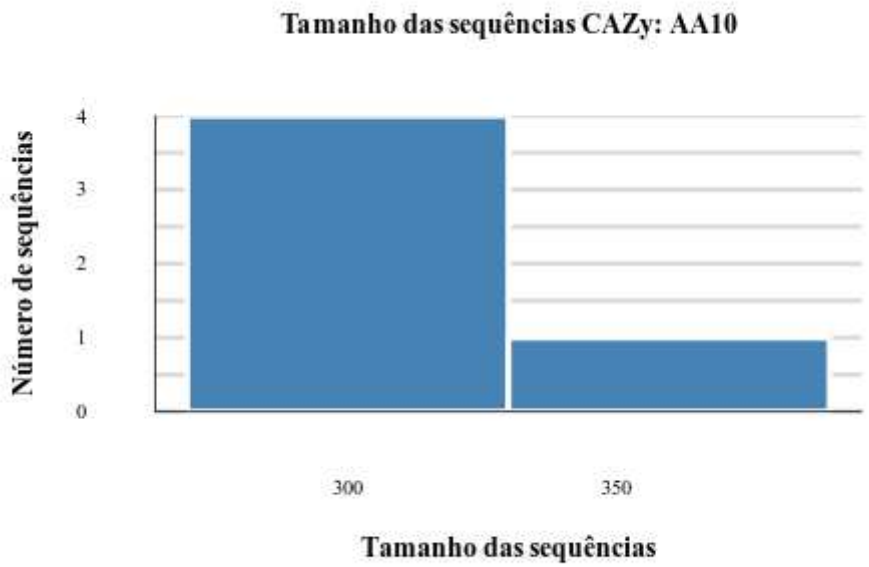

Figura 74 - Gráfico de tamanho de sequência AA10. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

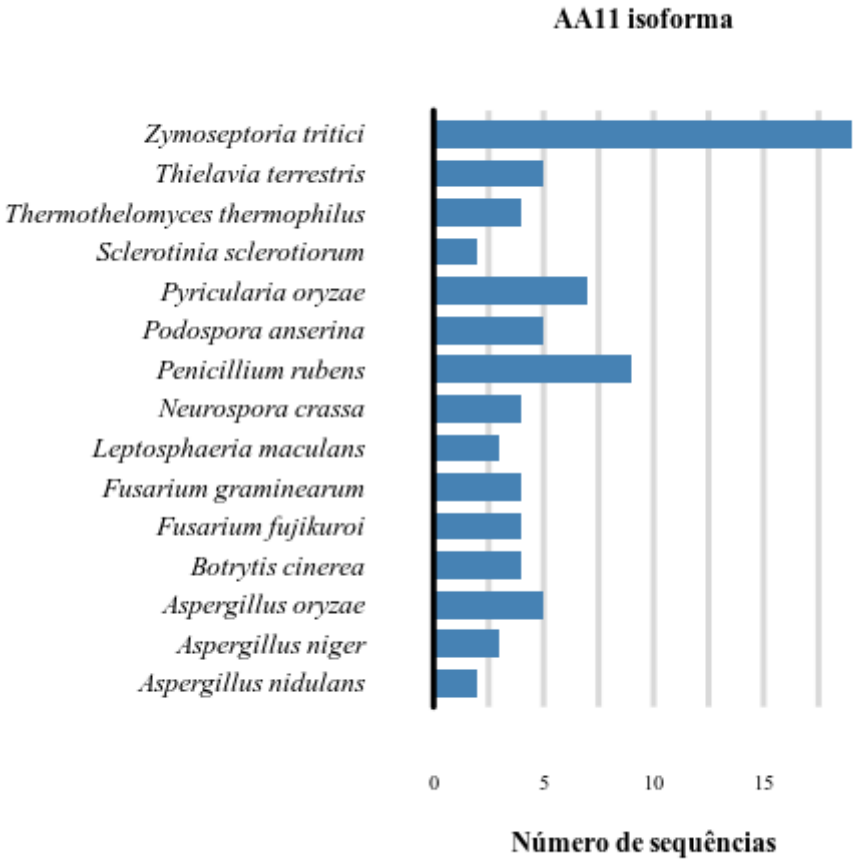

Figura 75 - Gráfico do número de isoformas da família AA11. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

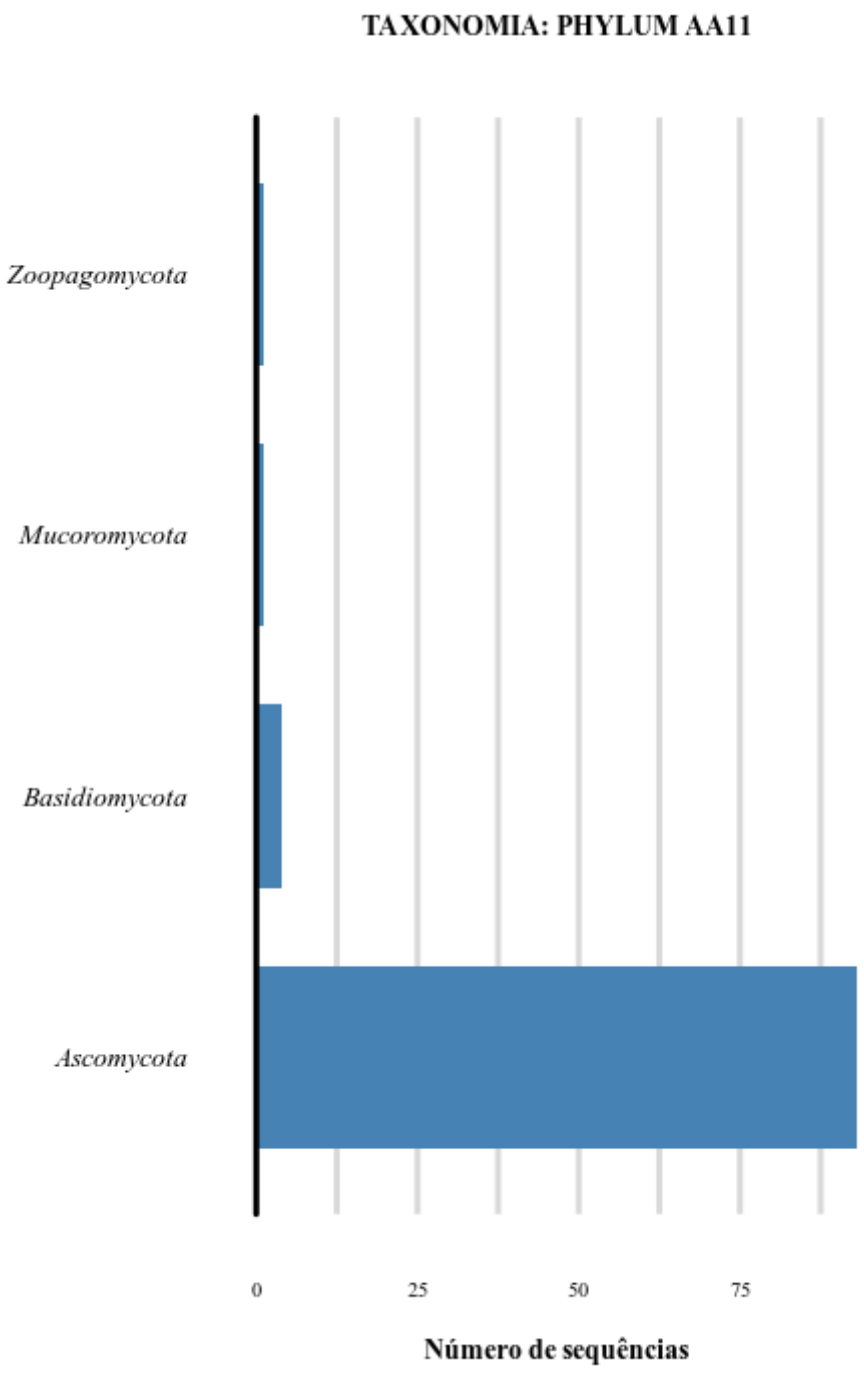

Figura 76 - Gráfico do número de sequências dentro dos filos da família AA11. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

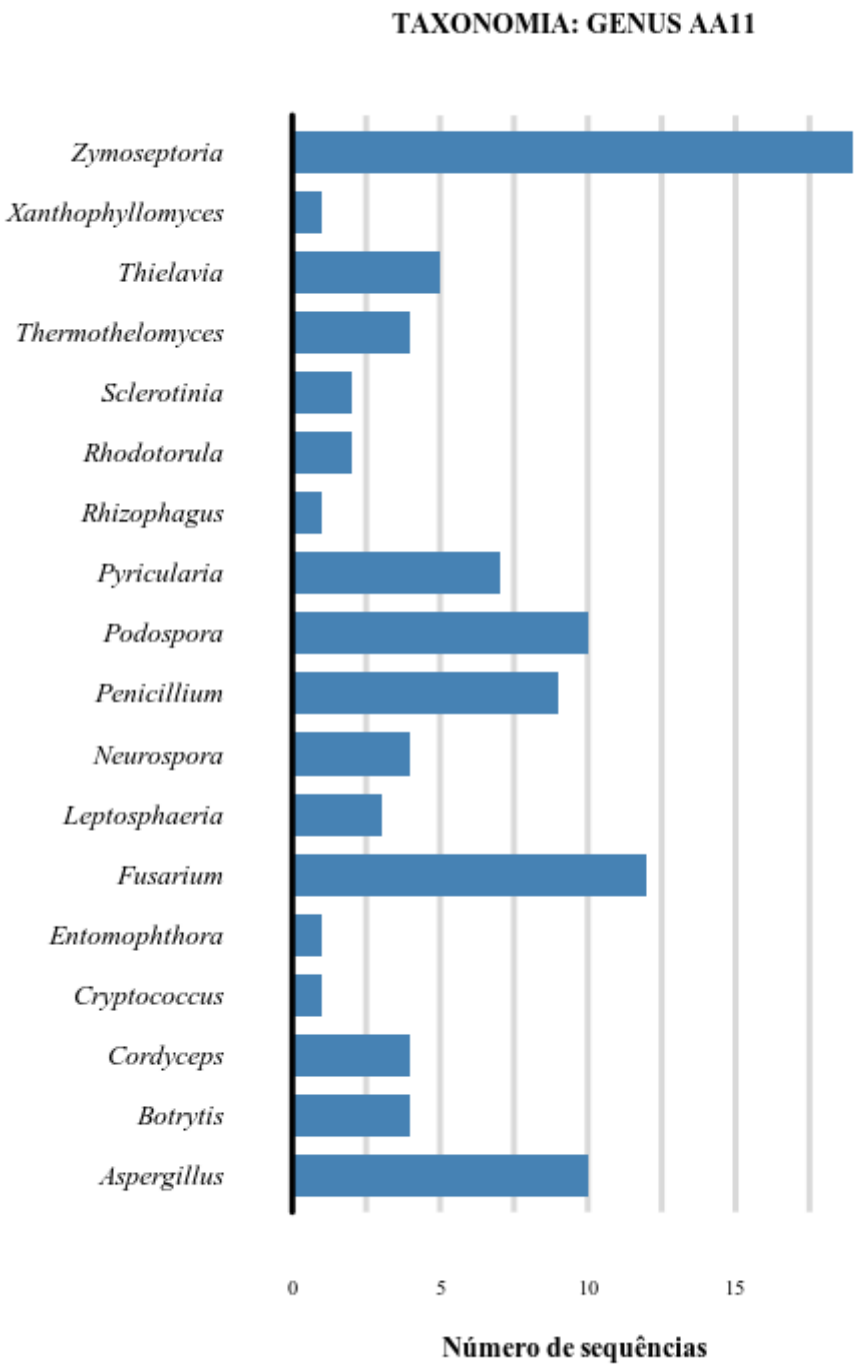

Figura 77 - Gráfico do número de sequências dentro dos gêneros da família AA11. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

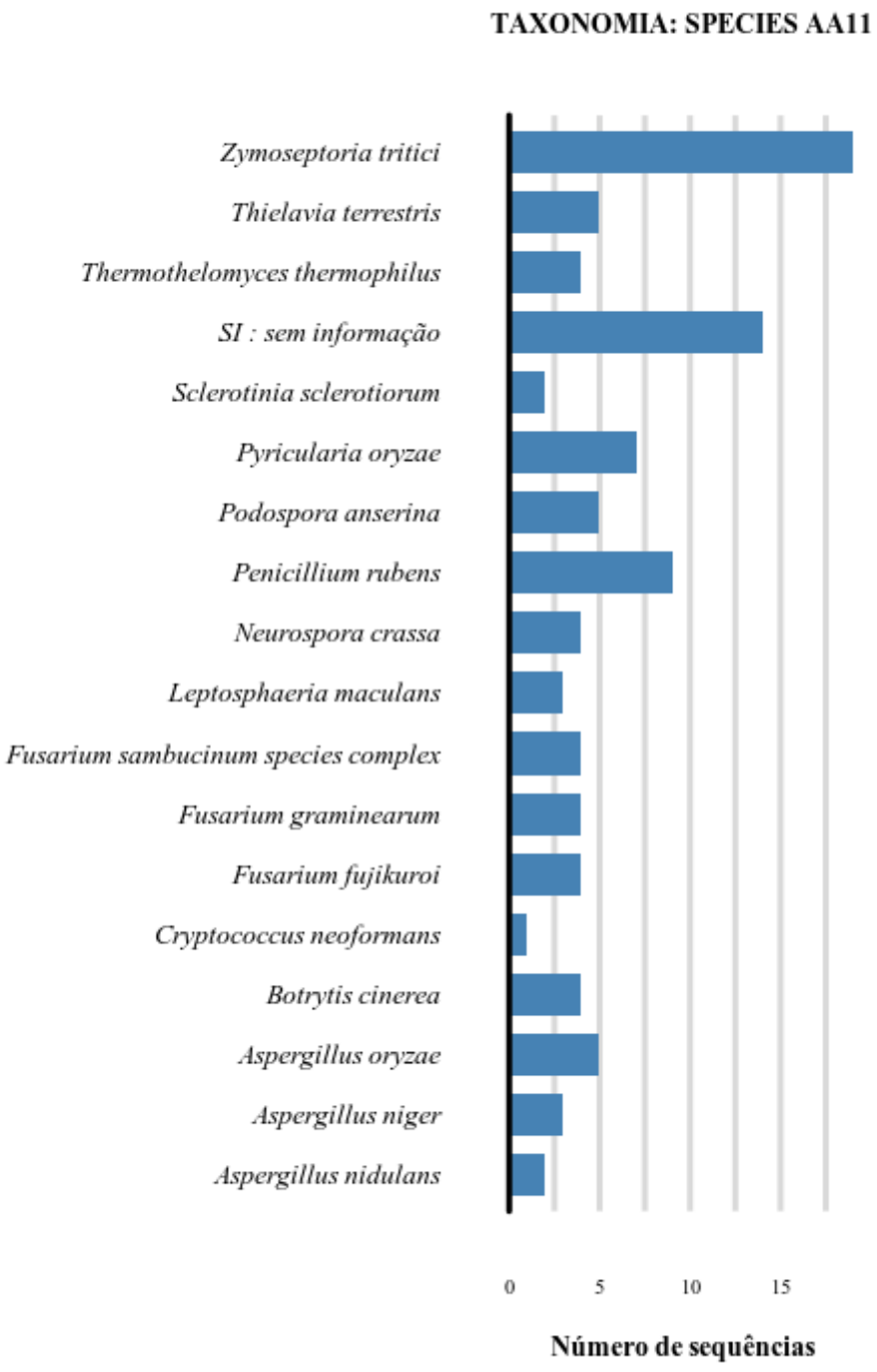

Figura 78 - Gráfico do número de sequências dentro das espécies da família AA11. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

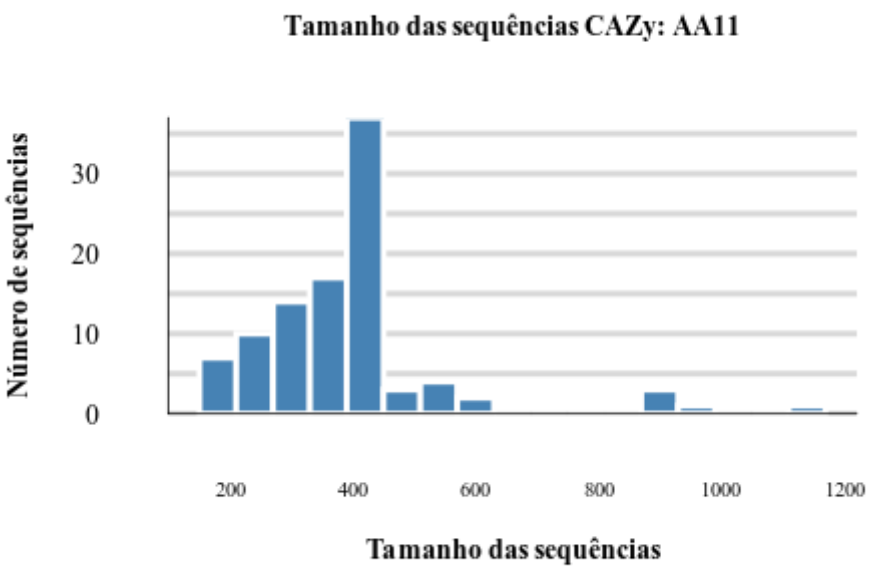

Figura 79 - Gráfico de tamanhos de sequência da família AA11. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

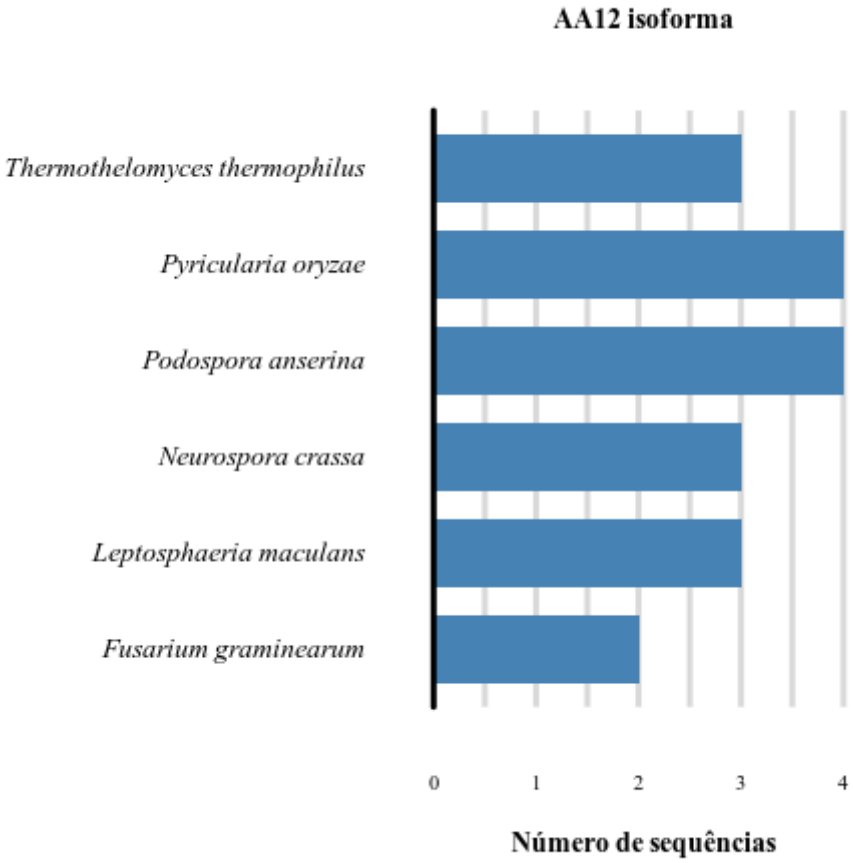

Figura 80 - Gráfico do número de isoformas da família AA12. O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

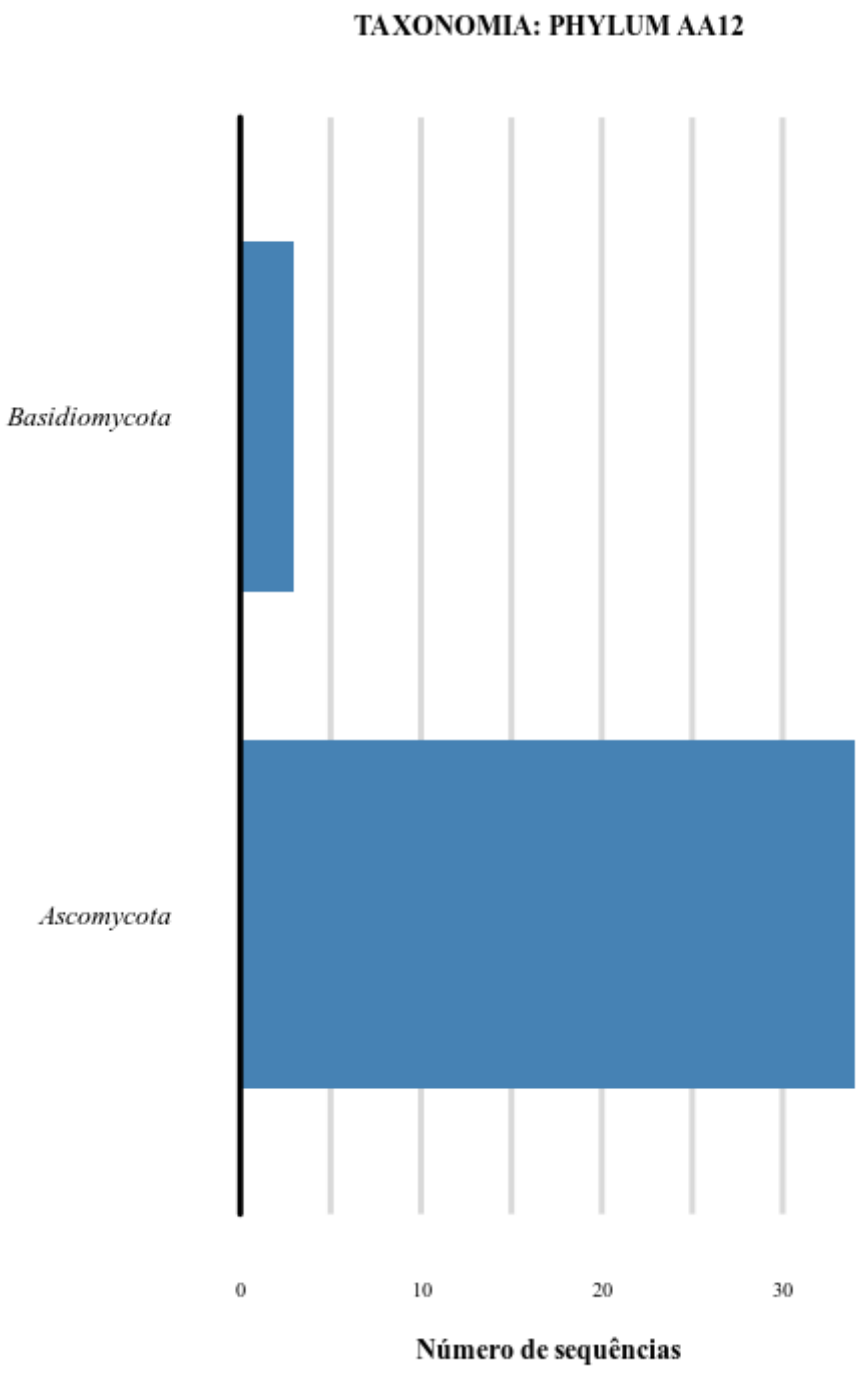

Figura 81 - Gráfico do número de sequências dentro dos filos da família AA12. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

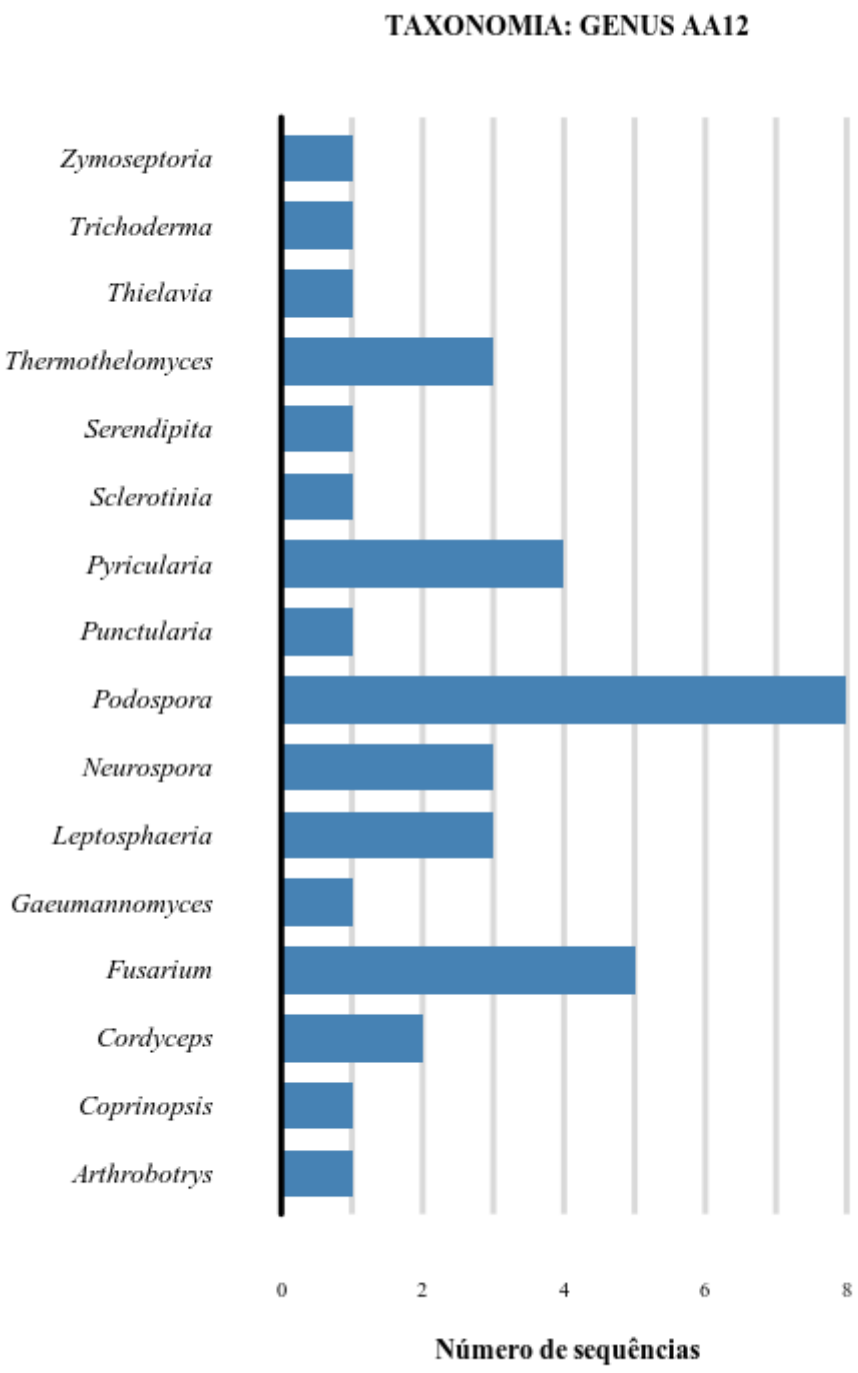

Figura 82 - Gráfico do número de sequências dentro dos gêneros AA12. O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

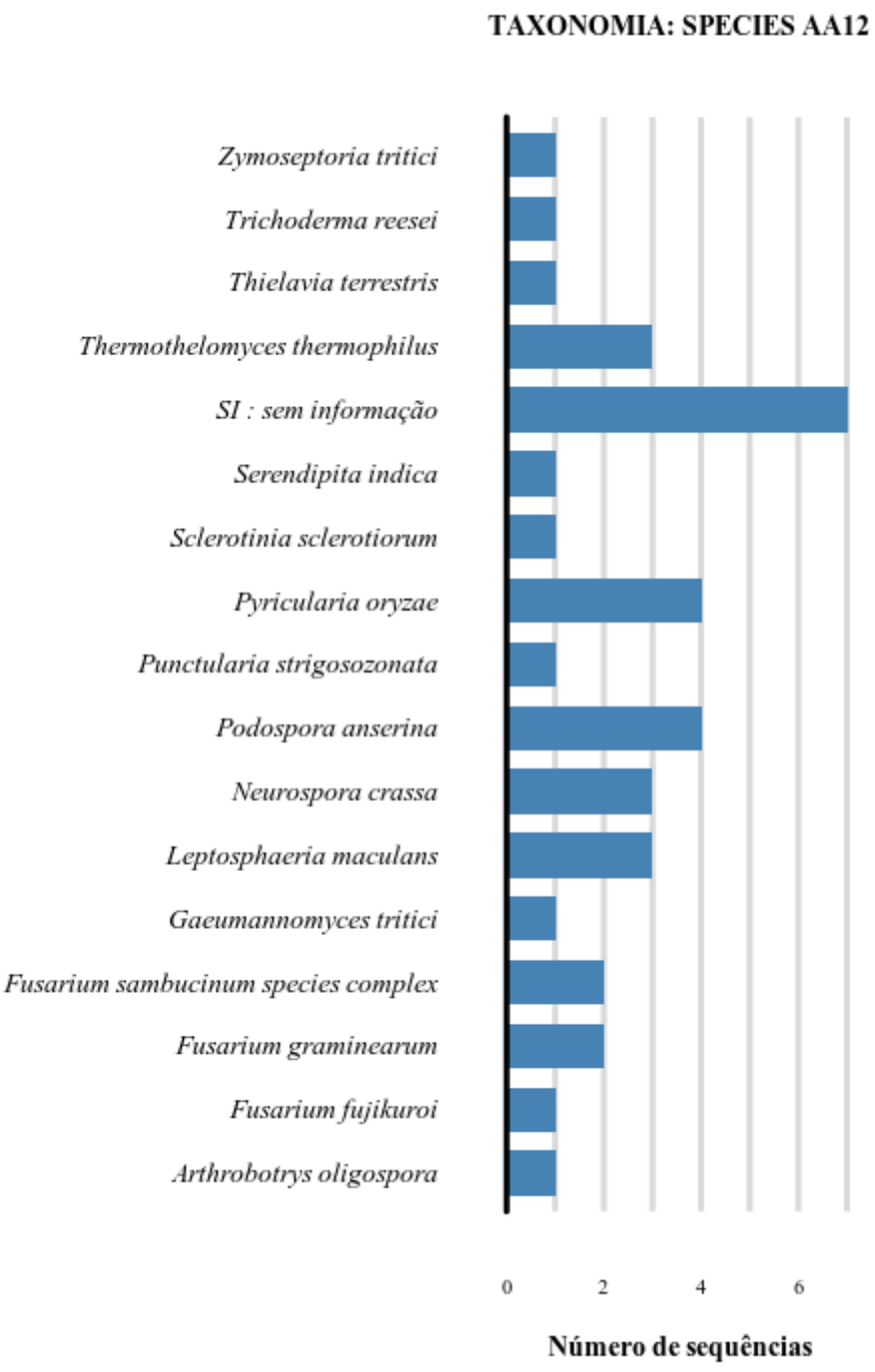

Figura 83 - Gráfico do número de sequências dentro das espécies AA12. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

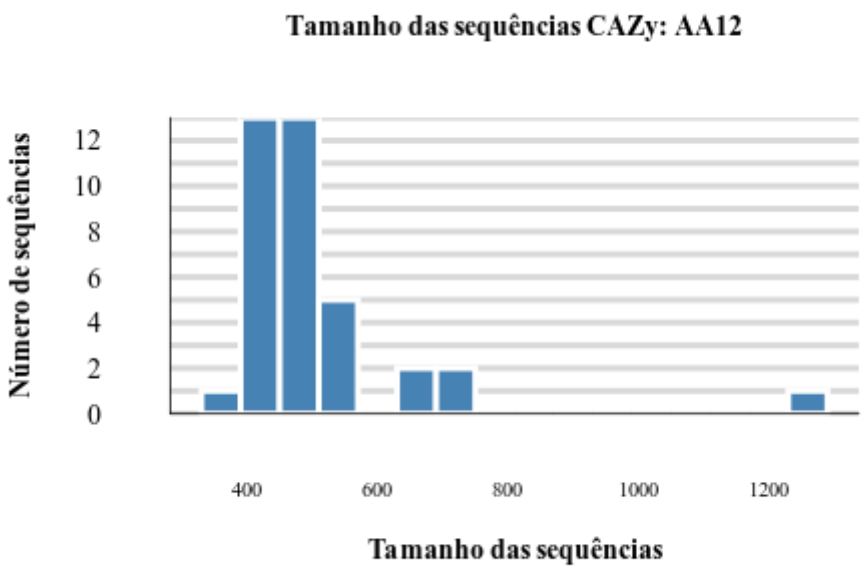

Figura 84 - Gráfico de tamanhos de sequência da família AA12.O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

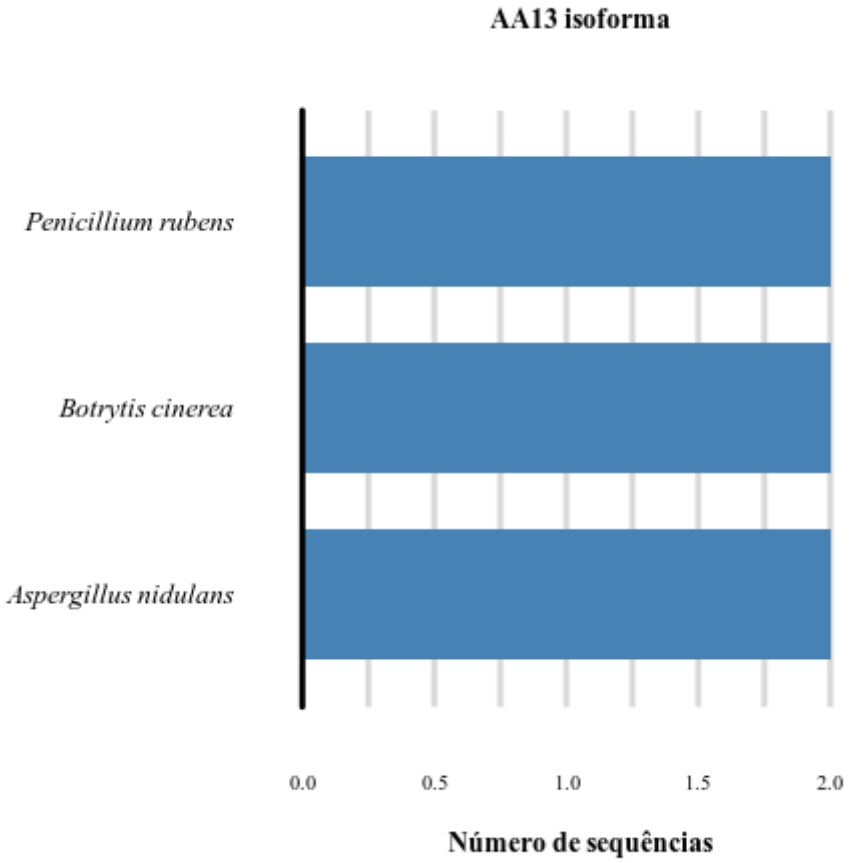

Figura 85 - Gráfico do número de isoformas da família AA13.O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

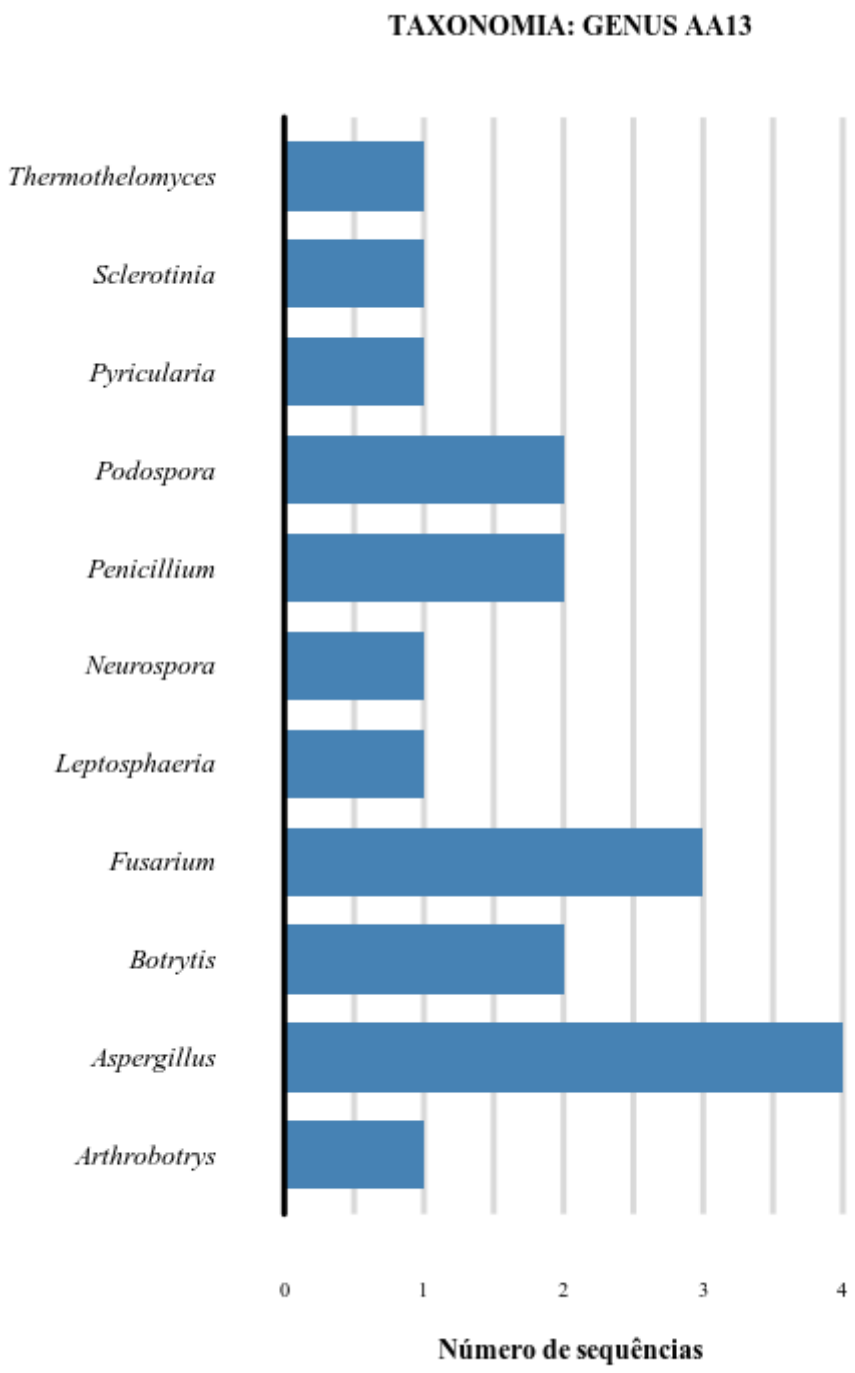

Figura 86 - Gráfico do número de sequências dentro dos gêneros da família AA13.O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

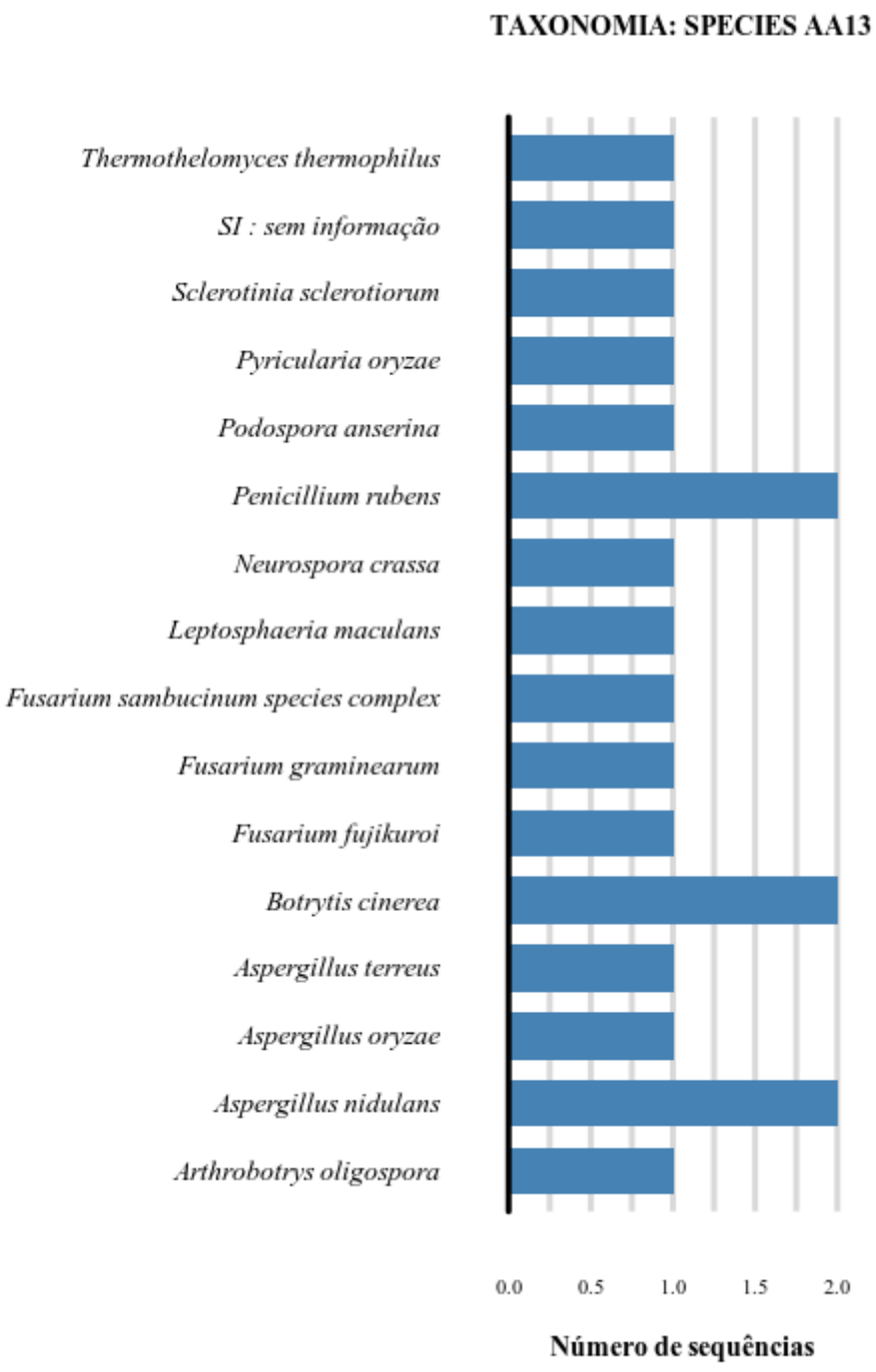

Figura 87 - Gráfico do número de sequências dentro das espécies da família AA13. O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

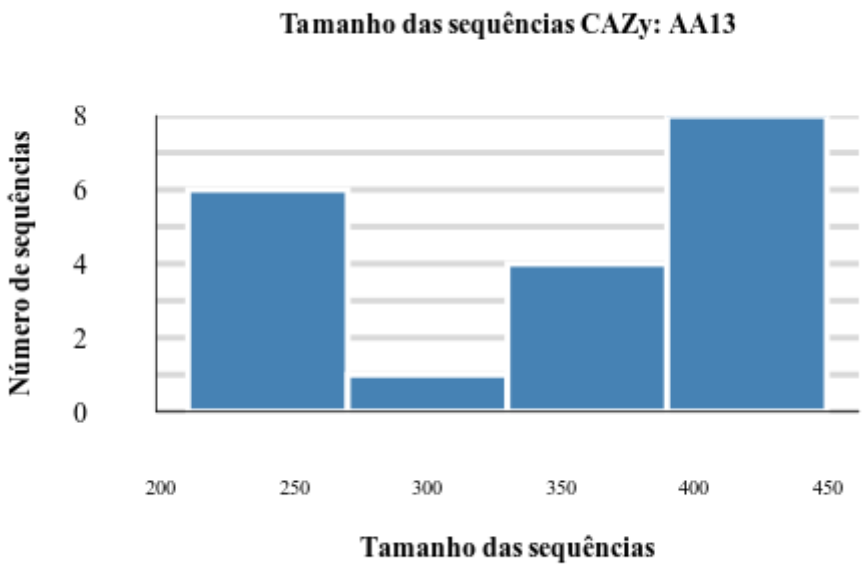

Figura 88 - Gráfico de tamanhos de sequência da família AA13. O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

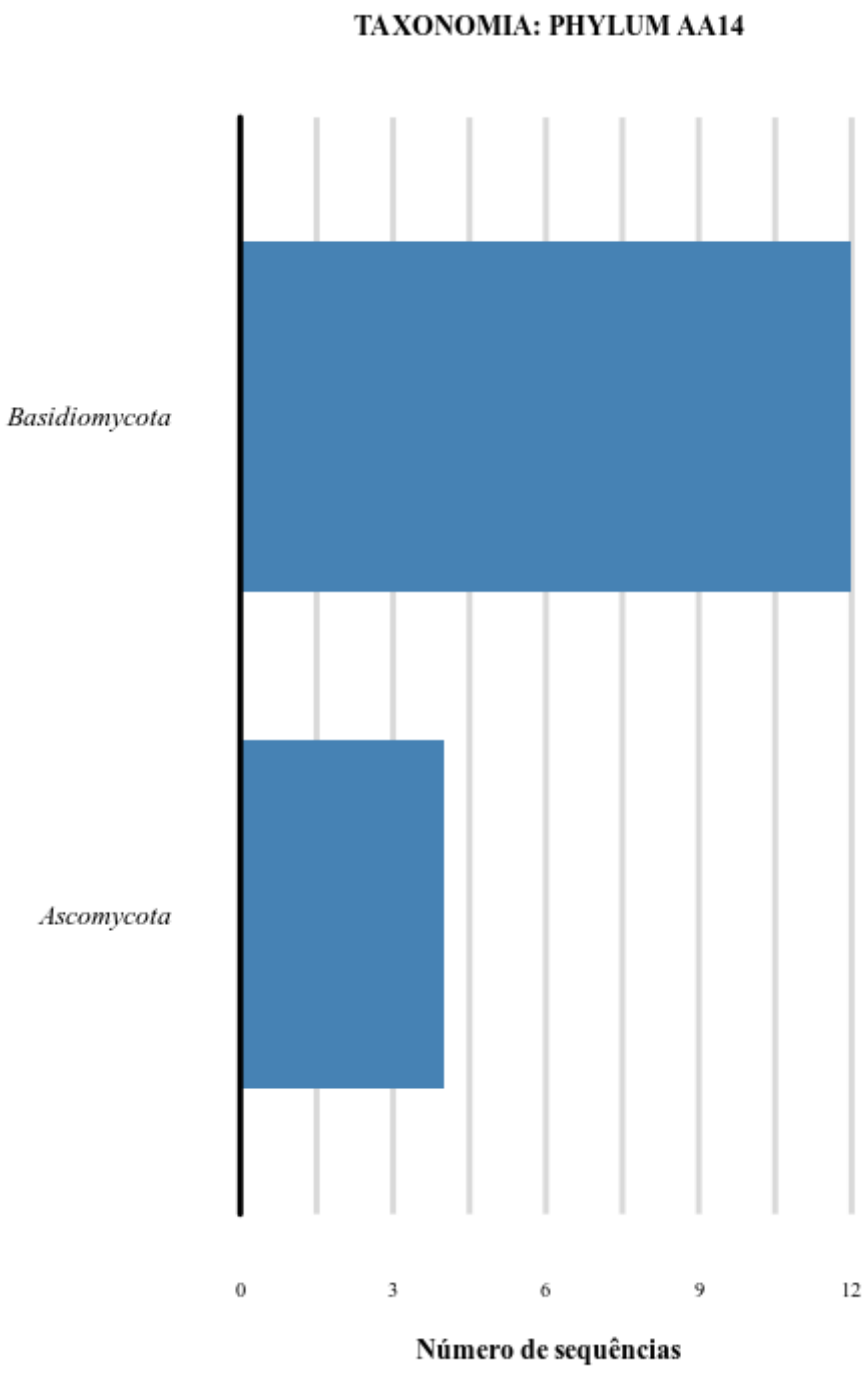

Figura 89 - Gráfico do número de sequências dentro dos filos da família AA14. O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

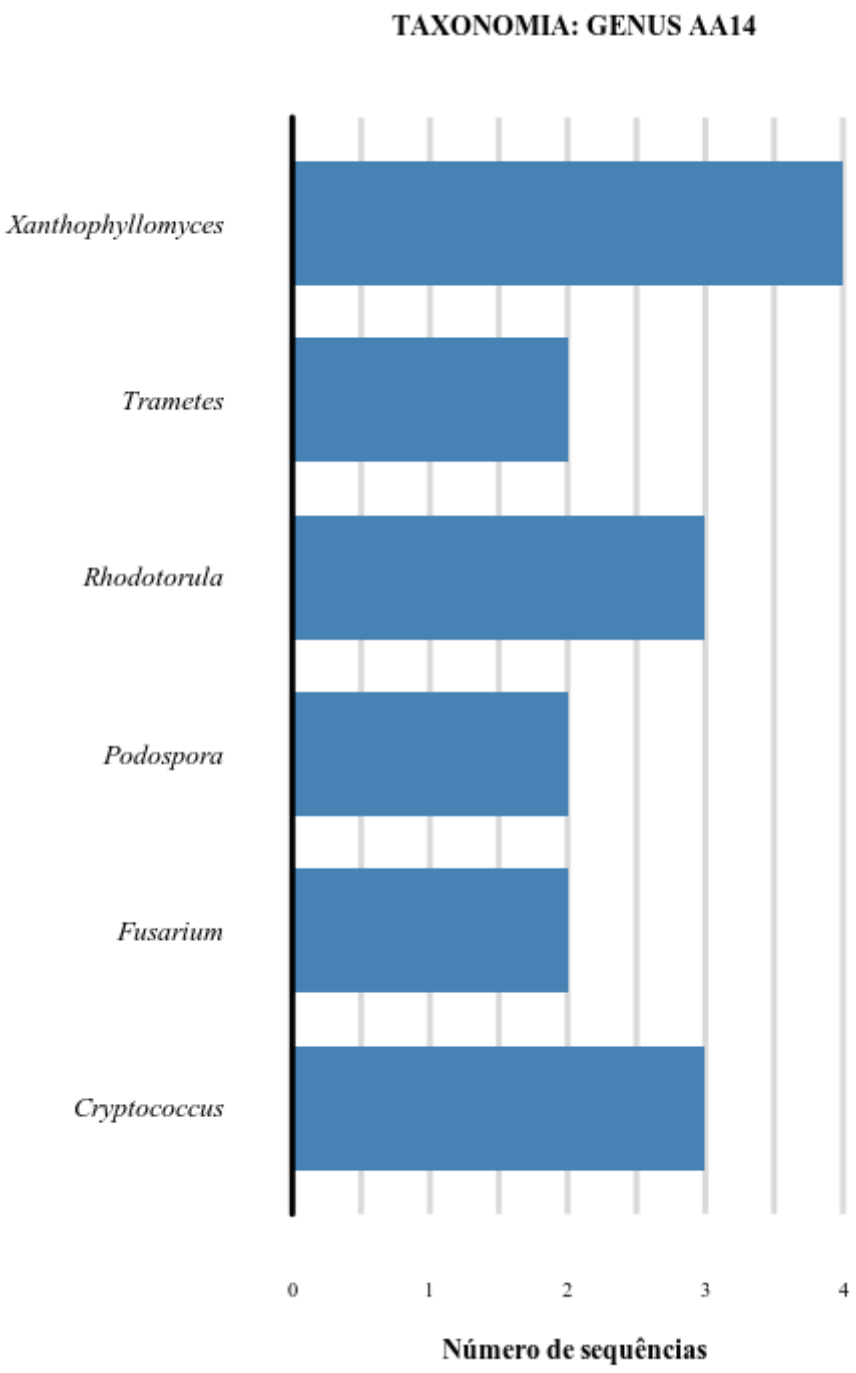

Figura 90 - Gráfico do número de sequências dentro dos gêneros da família AA14.O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

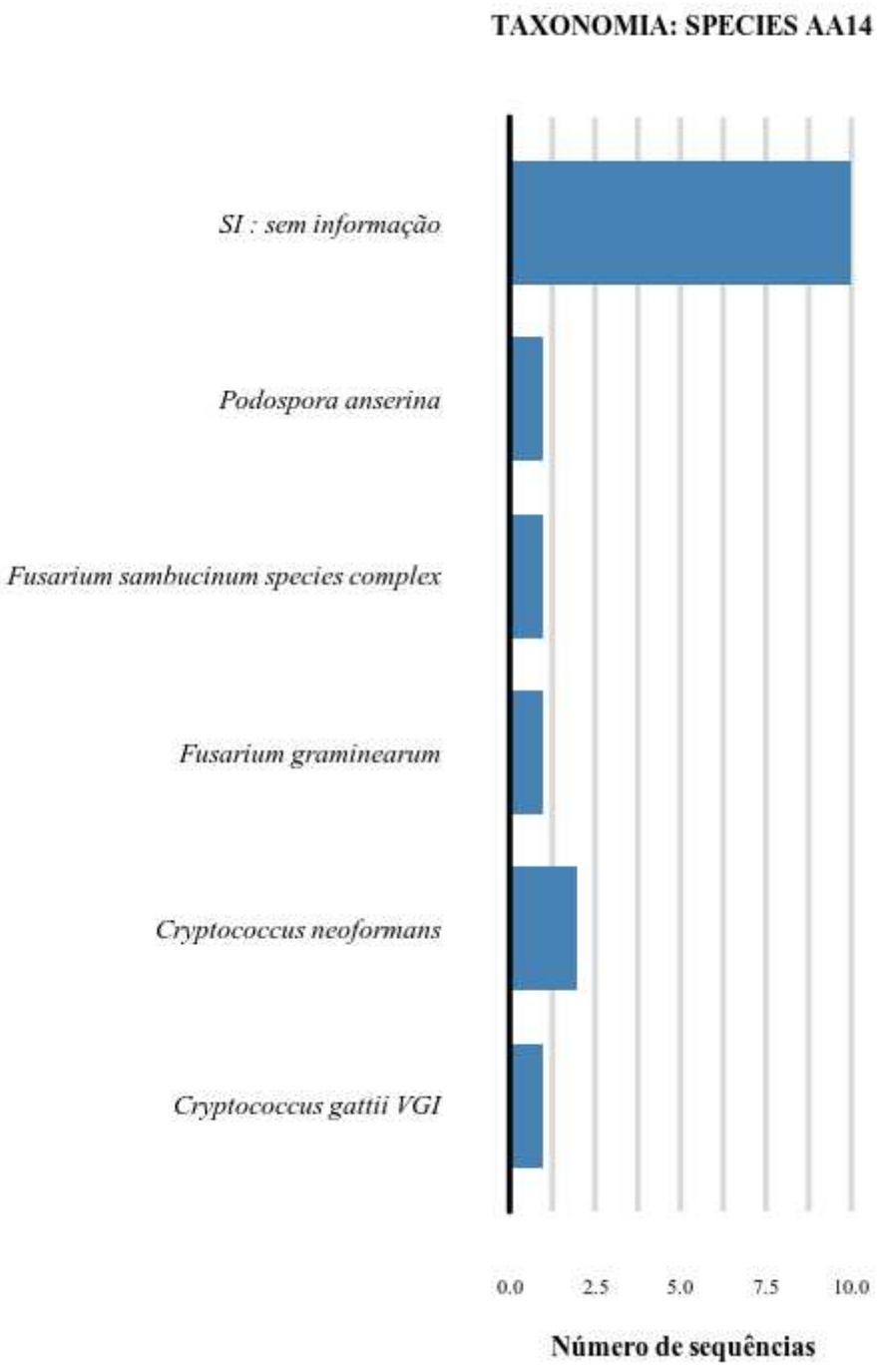

Figura 91 - Gráfico do número de sequências dentro das espécies da família AA14.O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

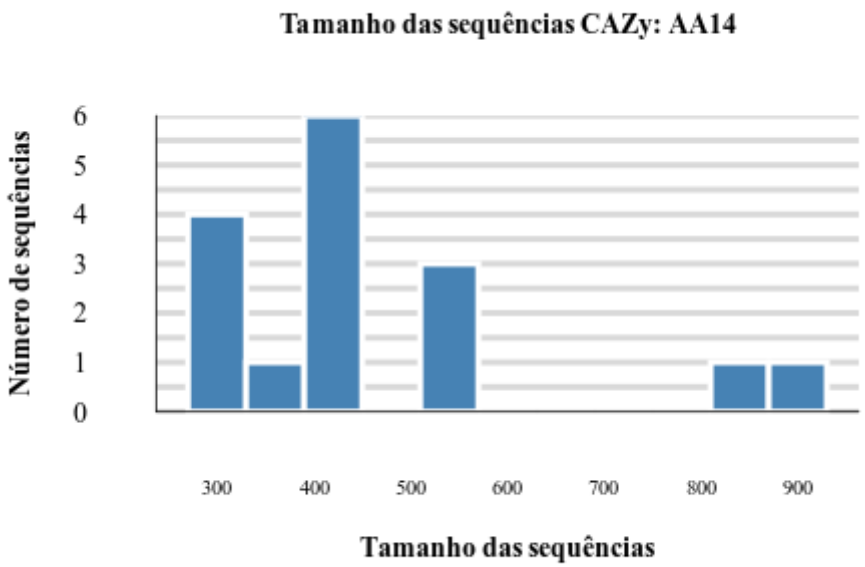

Figura 92 - Gráfico de tamanhos de sequência da família AA14.O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.

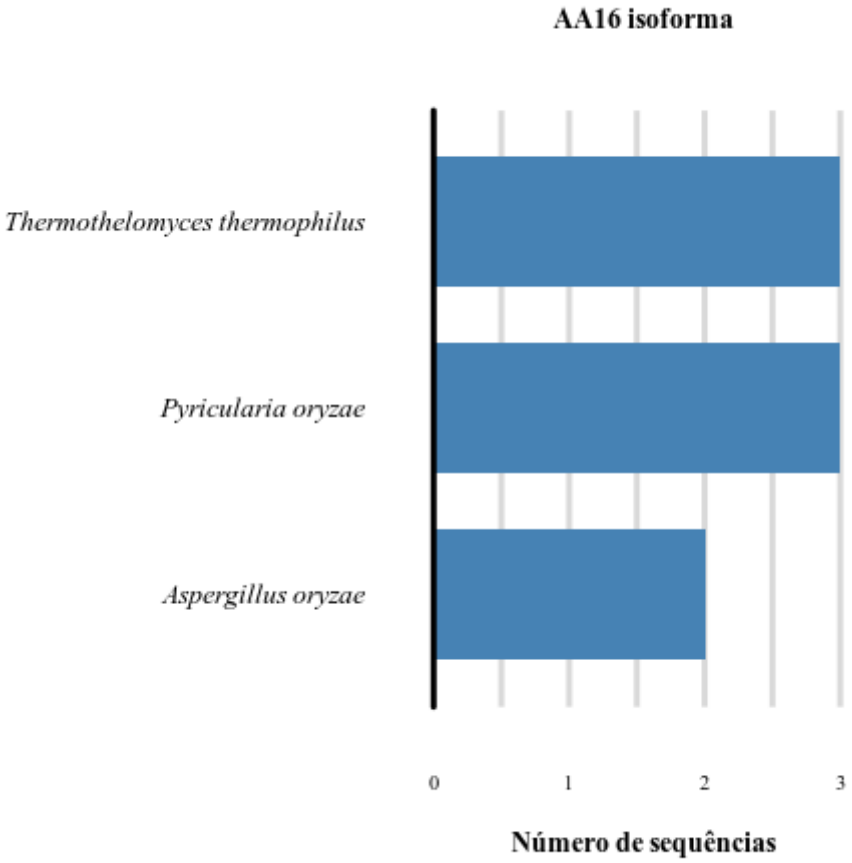

Figura 93 - Gráfico de isoforma da família AA16.O gráfico mostra o número de isoformas dentro de cada espécie. No eixo y se encontra o nome da espécie e no eixo x o número de isoformas.

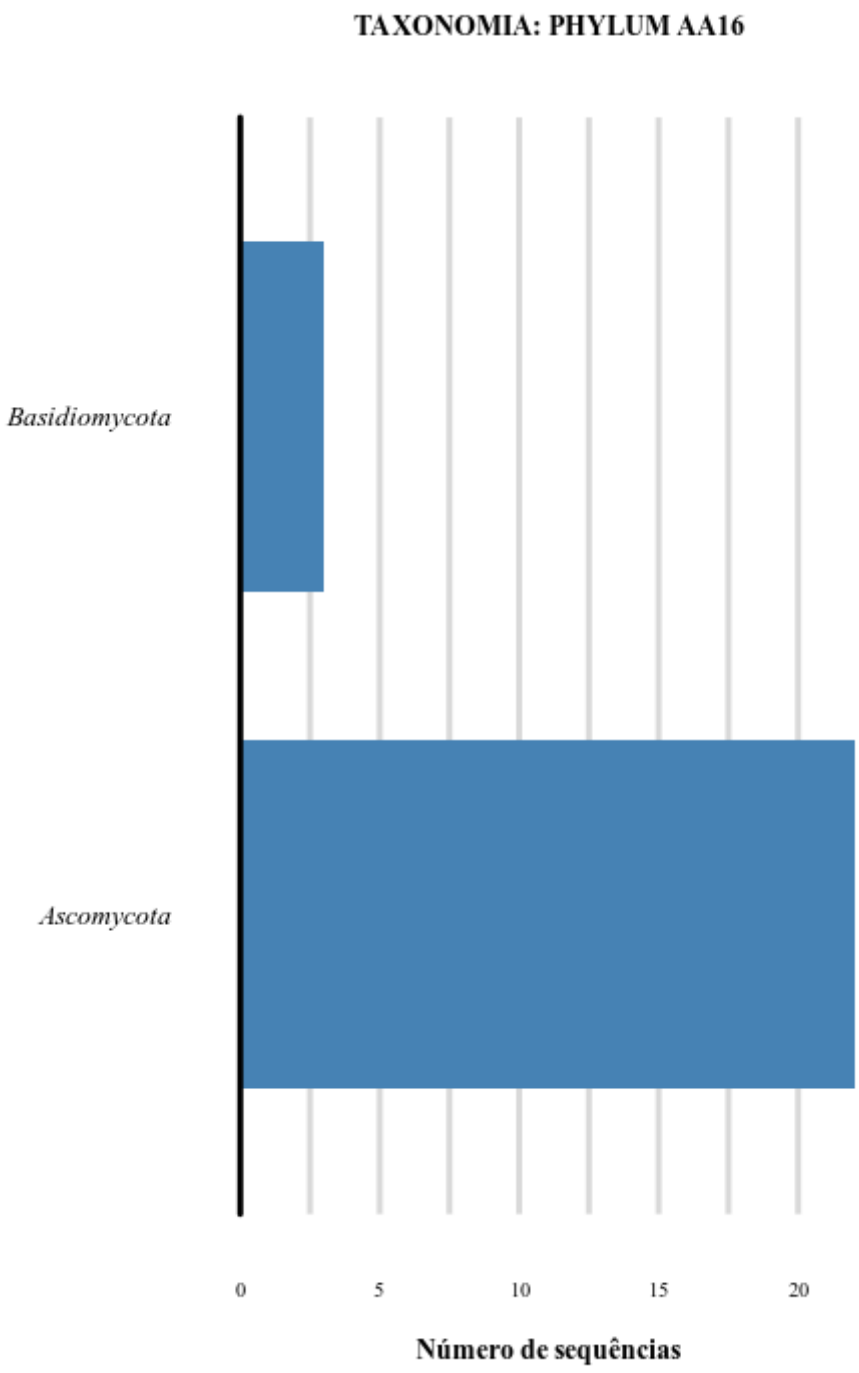

Figura 94 - Gráfico do número de sequências dentro dos filos da família AA16.O gráfico mostra o número de sequências dentro de cada filo. No eixo y se encontra o nome do filo e no eixo x o número de sequências.

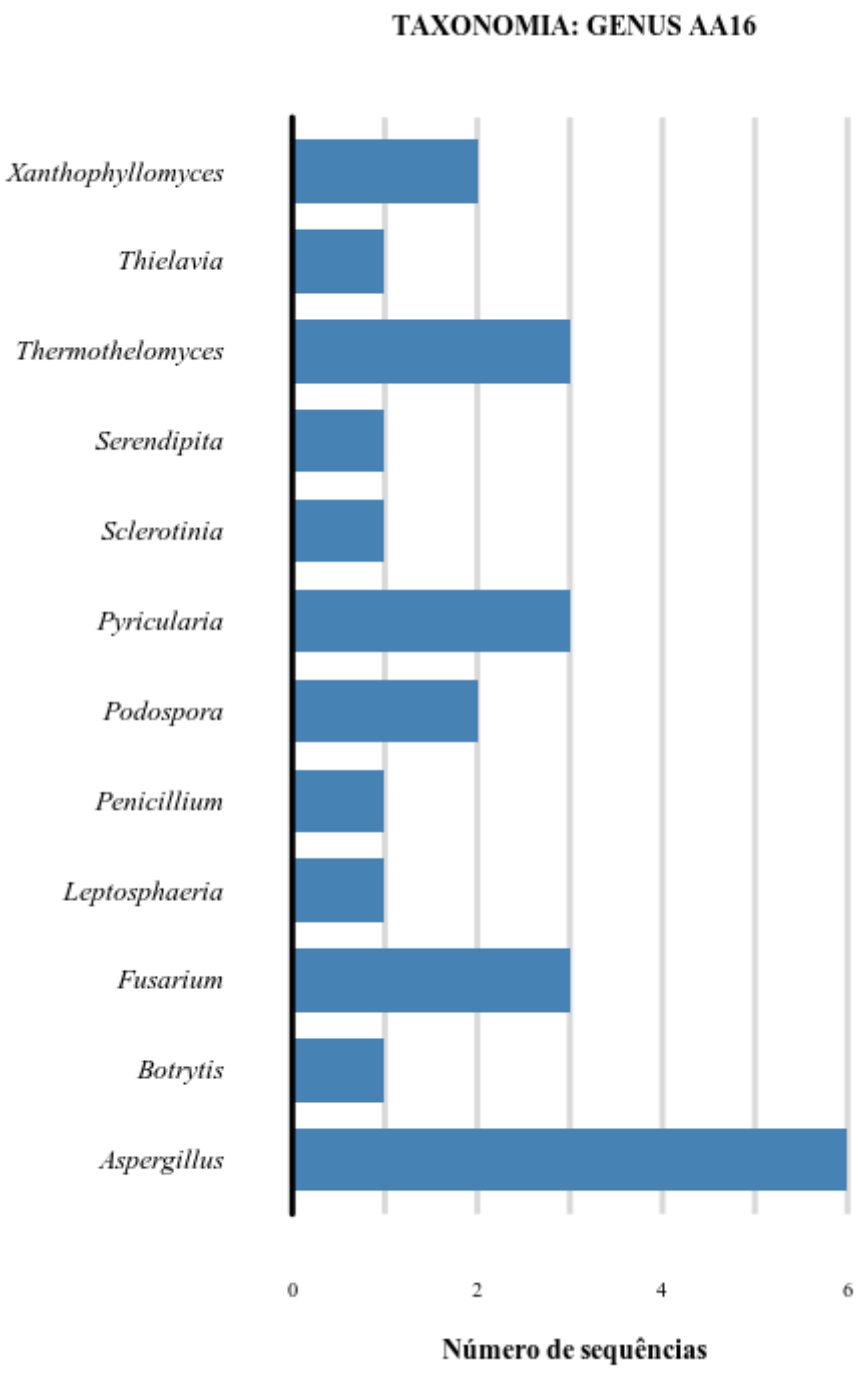

Figura 95 - Gráfico do número de sequências dentro dos gêneros da família AA16.O gráfico mostra o número de sequências dentro de cada gênero. No eixo y se encontra o nome do gênero e no eixo x o número de sequências.

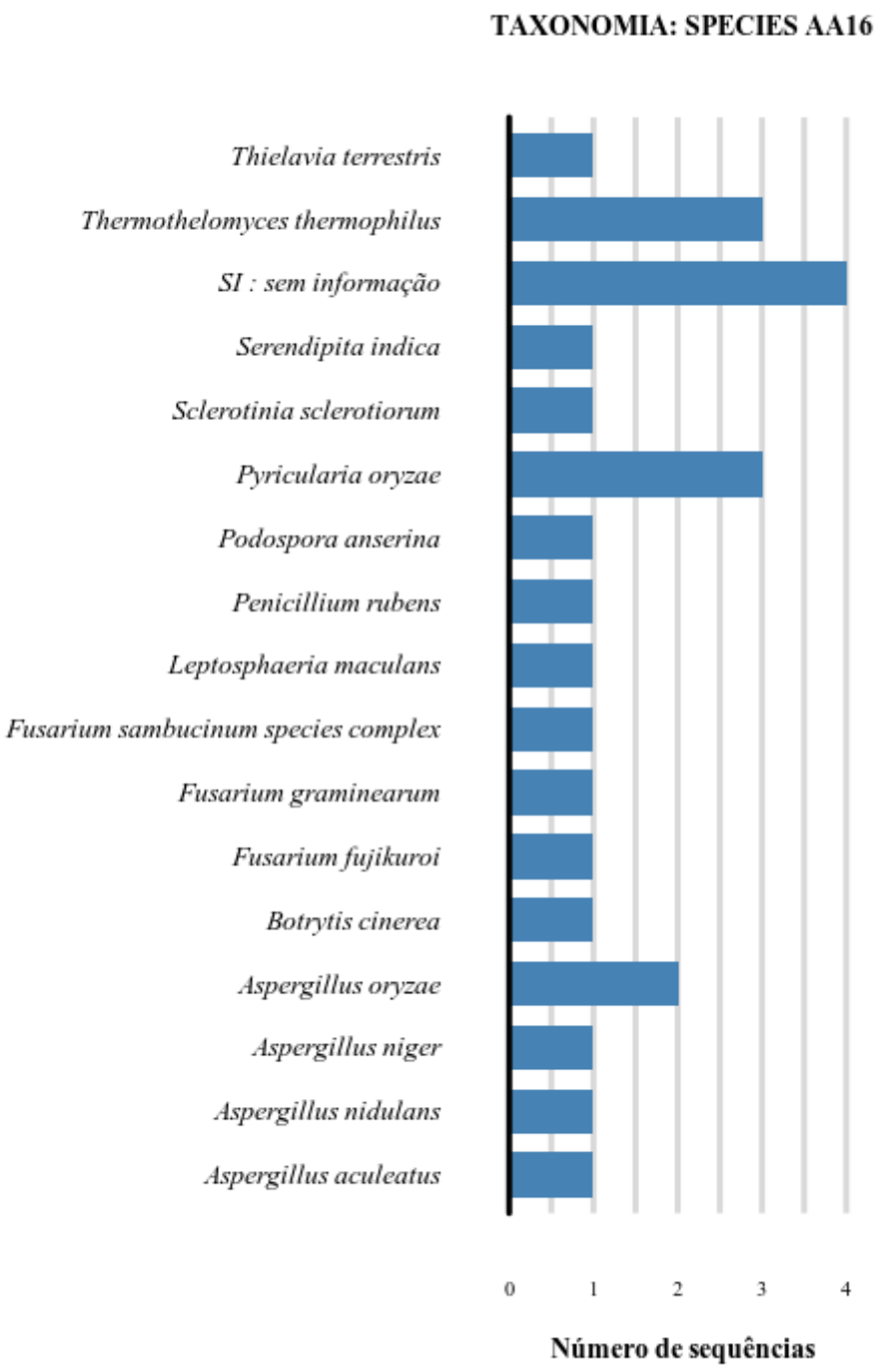

Figura 96 - Gráfico do número de sequências dentro das espécies da família AA16.O gráfico mostra o número de sequências dentro de cada espécie. No eixo y se encontra o nome do espécie e no eixo x o número de sequências.

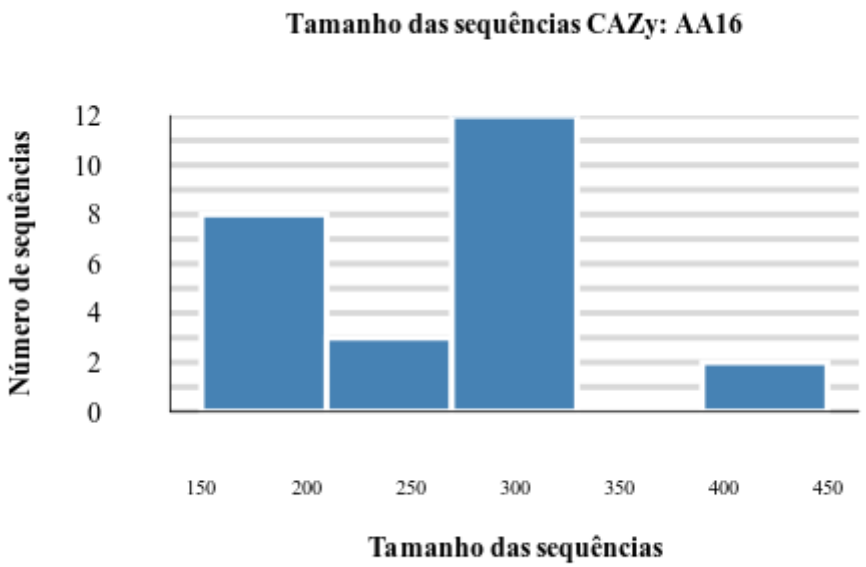

Figura 97 - Gráfico de tamanho de sequências da família AA16.O gráfico mostra o tamanho das sequências e quantas sequências estão naquela faixa de tamanho. No eixo y se encontra o número de sequências e no eixo x o tamanho da sequências.