HEURÍSTICAS HÍBRIDAS PARA O PROBLEMA DE COBERTURA DE ALVOS UTILIZANDO SENSORES

Marcos Vinícius Almeida Guimarães
Universidade Federal de Minas Gerais
Departamento de Ciência da Computação - R. Reitor Píres Albuquerque, ICEx - Belo Horizonte - MG
marcos.guimaraes@dcc.ufmg.br

Breno Costa Dolabela Dias
Ibmec/MG
R. Rio Grande do Norte, 300 - Funcionários, Belo Horizonte - MG
brenodolabela@gmail.com

Thiago Ferreira de Noronha
Universidade Federal de Minas Gerais
Departamento de Ciência da Computação - R. Reitor Píres Albuquerque, ICEx - Belo Horizonte - MG
tfn@dcc.ufmg.br

RESUMO

Redes de sensores vêm se tornando cada vez mais populares para fins de vigilância, segurança, monitoramento, dentre outros setores. Para se resolver o problema de posicionar sensores dentro de determinada área a fim de obter a cobertura máxima de alvos pode-se levar em consideração a utilização de técnicas que não permitirão que recursos sejam desperdiçados. Este trabalho apresenta oito heurísticas que propõem resolver o problema. Duas das quais estão presentes na literatura, duas são adaptações de metaheurísticas já existentes e as quatro restantes são modificações das anteriores que, através de experimentos computacionais com diferentes cenários, mostram-se superiores às demais.

PALAVRAS CHAVE. Redes de sensores. Cobertura máxima com o menor número de sensores. Metaheurísticas.

Área principal: Heurísticas.

ABSTRACT

Sensor networks are becoming increasingly popular for surveillance, security, monitoring, among others ends. To solve the problem of positioning sensors within a given area and getting the maximum coverage of the targets, one can take into account the use of techniques that will not allow resources to be wasted. This paper presents eight heuristics that can be used to solve this problem. Two of them are present in the literature, two are adaptations of existing metaheuristics and the remaining four are modifications of the previous ones that, through computational experiments with different scenarios, have better performance than the others.

KEYWORDS. Sensor networks. Maximum coverage with minimum sensors. Metaheuristics.

Main area: Heuristics.

1. Introdução

Redes de sensores visuais vêm sendo alvo de pesquisas nos últimos anos devido ao grau de aplicabilidade em cenários reais. Conforme os preços de sensores de transmissão de vídeo vão reduzindo, mais viável é a utilização dos mesmos em cenários de vigilância, monitoramento ambiental, dentre outros [Alkyildiz et al., 2007], [Soro e Heinzelman, 2009] e [Rashid e Rehmani, 2016].

Uma rede de sensores visuais, como definida por [Hafsa et al., 2016], pode ser descrita como um conjunto de alvos que devem ser monitorados por um segundo conjunto de sensores. Para que o aproveitamento máximo do sistema seja alcançado deve-se levar em consideração critérios como o consumo mínimo de bateria. Por isso, o objetivo principal do problema é utilizar o menor número de sensores para cobrir o maior número possível de alvos.

Redes de sensores visuais podem ser classificadas em sistemas super-provisionados (over-provisioned systems), onde o número de sensores disponíveis é suficiente para cobrir todos os alvos, logo o objetivo principal visa minimizar o número de sensores utilizados e, ao mesmo tempo, maximizar o número de alvos cobertos. Já nos sistemas infra-provisionados (under-provisioned systems), o número de sensores disponíveis não é suficiente para cobrir todos os alvos, logo tem-se como objetivo maximizar a cobertura de alvos, independente do número de sensores sendo utilizados.

O processo de desenvolvimento de heurísticas para resolver o problema pode ser dividido em duas abordagens: abordagem orientada aos sensores e abordagem orientada aos alvos [Ai e Abouseid, 2006], [Fusco e Gupta, 2009], [Munishwar e Abu-Ghazaleh, 2011] e [Munishwar e Abu-Ghazaleh, 2013]. A primeira tem como objetivo maximizar o número de alvos que podem ser cobertos pelos sensores, ou seja, deve-se cobrir o maior número de alvos possíveis com eles e a segunda visa cobrir primeiramente os alvos que são cobertos pelo menor número de sensores.

1.1. Caracterização do problema

O problema citado acima, conhecido como o problema de máxima cobertura com o menor número de sensores (MCMS), é uma generalização do problema de cobertura de conjuntos e pode ser definido como, dado um conjunto de alvos $A = \{a_1, a_2, ..., a_m\}$ a serem cobertos e um conjunto de sensores ou câmeras $S = \{s_1, s_2, ..., s_n\}$ com p possíveis orientações a serem aleatoriamente posicionadas em um plano bidimensional, tem-se como objetivo usar o menor número possível de sensores para cobrir o maior número possível de alvos.

O valor de p pode ser obtido através de um ângulo $0 < \theta \leq 360$ que representa o campo de visão dos sensores. Por exemplo, se $\theta = \pi/4$ teremos 8 possíveis valores para p que representam 8 possíveis orientações para um sensor qualquer s_i. ϕ_{ij} representa um conjunto de alvos que são cobertos por um sensor i quando sua orientação é j.

A coleção de subconjuntos $F = \{\phi_{ij}, 1 \leq i \leq n, 1 \leq j \leq p\}$ pode ser obtida após calcular a relação entre os campos de visão dos sensores e os alvos a serem cobertos, onde cada elemento de ϕ_{ij} é um subconjunto de A.

Tem-se como objetivo escolher um subconjunto desses n sensores e a orientação de cada um a fim cobrir o maior número de alvos possível. De acordo com [Ai e Abouseid, 2006] e [Cai et al., 2007] esse problema é classificado como NP-completo.

1.2. Motivação e Objetivos

Como foi dito na subseção anterior, a versão de decisão do problema MCMS pertence à classe NP-completo, o que torna sua versão de otimização NP-difícil. Como não existem formas de se desenvolver algoritmos em tempo polinomial para encontrar soluções para problemas NP-completos e NP-difíceis, justifica-se o uso de heurísticas que não garantirão soluções ótimas, porém serão executadas em uma margem de tempo controlada.

O trabalho em questão tem como objetivo a implementação de quatro heurísticas da literatura e apresenta quatro modificações das mesmas. Experimentos foram realizados a fim de comparar as oito implementações e os resultados foram mostrados.

Tem-se como justificativa para a obtenção desses resultados o fato de que, às vezes, pequenas modificações em trabalhos ou algoritmos já existentes podem acarretar em grandes melhoras nos resultados encontrados.

1.3. Roteiro

A seção seguinte mostrará trabalhos relacionados ao abordado nesse artigo e apresentará os 4 algoritmos presentes na literatura. A seção 3 apresentará as 4 modificações propostas para os algoritmos presentes na seção 2. A seção 4 descreverá os experimentos computacionais que foram realizados e mostrará os resultados em forma de gráficos. Por último, a seção 5 trará conclusões a respeito dos resultados obtidos na realização desse trabalho.

2. Trabalhos relacionados

Um número notável de pesquisas que envolvem redes de sensores vem sendo feitas na última década. Dentre eles, [Chakrabarty et al., 2002] mostraram problemas de posicionamento ótimo assumindo que o campo no qual os sensores seriam posicionados seria uma grade de pontos e que os mesmos só poderiam ser posicionados nesses pontos. Foi proposto uma formulação de Programação Linear Inteira para resolver pequenas instâncias desse problema e uma abordagem de divisão e conquista para resolver grandes instâncias.

Em [Hörster e Lienhart, 2006] foi proposto um algoritmo guloso que leva em consideração a atribuição de critérios que atribuem prioridades aos sensores na hora de serem posicionados. Tais critérios levam em consideração o custo de posicionar tais sensores e o número de alvos que eles cobrem.

Também pode ser encontrado em [Ai e Abouseid, 2006] uma formulação de programação linear inteira para resolver o problema. Tal formulação tem como objetivo maximizar o número de alvos cobertos e ao mesmo tempo minimizar o número de sensores que são usados. Também foram apresentadas duas heurísticas gulosas para resolver grandes instâncias do problema levando em consideração que a primeira formulação não alcança tais objetivos.

Ao se considerar cenários onde o posicionamento dos sensores é feito de forma aleatória, deve-se manter o mínimo de sensores ativos devido a limitações de energia. Assim sendo, [Munishwar e Abu-Ghazaleh, 2011] apresentaram uma abordagem para esse cenário que atribui critérios de prioridades para os sensores e os orienta na direção em que cobrem o maior número de alvos.

Em [Hafsa et al., 2016] foram apresentadas seis abordagens para resolver o problema utilizando diferentes critérios. Foram utilizadas a formulação de programação linear inteira e umas das heurísticas apresentadas em [Ai e Abouseid, 2006], a heurística mostrada em [Munishwar e Abu-Ghazaleh, 2011] e três outras que foram desenvolvidas utilizando critérios gulosos. Ao final foram realizados experimentos que proporcionaram aos autores critérios para que os algoritmos pudessem ser comparados.

2.1. Descrição de duas heurísticas existentes para o MCMS

Nessa subseção serão apresentadas duas heurísticas que foram desenvolvidas em [Hafsa et al., 2016] e que serviram de base para que o trabalho proposto neste artigo fosse realizado. Ambas as heurísticas são orientadas aos alvos e dá-se prioridade aqueles que são cobertos por apenas um sensor.

2.1.1. Heurística Gulosa Orientada aos Alvos (GTOH)

Como citado anteriormente, em GTOH primeiramente procura-se cobrir apenas os alvos que são cobertos por apenas um sensor. Tal abordagem é possível graças a critérios de peso que são dados a cada alvo. Tais critérios podem ser descritos da seguinte forma:

$$\mathrm{w}_j = \begin{cases} 1 & \textit{se o alvo } t_j \textit{ for coberto por apenas um sensor} \\ 0 & \textit{caso contrário} \end{cases}$$

Considere um cenário onde N representa o número total de sensores disponíveis. Como um alvo pode ser coberto no máximo por N sensores, o pior peso possível é N. Essa abordagem permite que todos os alvos que são cobertos por mais de um sensor sejam tratados igualmente,

dando prioridade aos que são cobertos por apenas um sensor. Depois que todos os alvos solitários são cobertos, são atribuídos valores (chamados aqui de ranking) para cada campo de visão dos sensores. O critério de atribuição desse valor segue a seguinte fórmula:

$$R_{ik} = \sum_{j=1}^{M} \left\{ \frac{1}{w_i} : t_j \in F(\langle s_i,p_k \rangle) \right\} i \in [1,N]; k \in [1,Q];$$

R_{ik} representa o ranking de um campo de visão p_k de um sensor s_i. $F(\langle s_i,p_k \rangle)$ representa o conjunto de alvos localizado no campo de visão p_k de um sensor s_i. Depois de atribuir um ranking para cada campo de visão dos sensores, o campo de visão que possui o maior ranking é selecionado e orientado.

Algorithm 1 Heurística Gulosa Orientada aos Alvos (GTOH)

Input: C = conjunto de sensores; T = conjunto de alvos descobertos; P = conjunto de campos de visão dos sensores

Output: Z = Conjunto sensores com orientação definida

1: **repeat**
2: Computar w_j para cada $t_j \in T$, e R_{ik} para cada $p_k \in P$ de $c_i \in C$
3: Construir um conjunto T_1 contendo somente os alvos solitários de T
4: **if** $T_1 \neq \emptyset$ **then**
5: $S = \emptyset$
6: **for all** $c_j \in C$ **do**
7: **for all** $p_l \in P$ **do**
8: $T_{jl} = F(\langle c_j,p_l \rangle) \cap T_1$
9: $S = S \cup T_{jl}$
10: **end for**
11: **end for**
12: Encontrar o conjunto T_{ik} em S cujo tamanho é máximo
13: $Z = Z \cup \langle c_i,p_k \rangle$
14: $C = C - c_i$
15: $T = T - T_{ik}$
16: **else**
17: Encontrar $\langle c_m,p_n \rangle$ que tem o maior ranking R_{mn} dentre todos os disponíveis
18: Suponha que T_3 é o conjunto de alvos que são cobertos por $\langle c_m,p_n \rangle$
19: $Z = Z \cup \langle c_m,p_n \rangle$
20: $C = C - c_i$
21: $T = T - T_3$
22: **end if**
23: **until** $S = \emptyset$ ou $T = \emptyset$

GTOH recebe como parâmetros de entrada o conjunto de sensores, o conjunto de alvos e o conjunto de campos de visão dos sensores a serem utilizados. A linha 2 calcula o peso de todos os alvos e o ranking de todos os campos de visão dos sensores. A linha 3 é responsável por encontrar o subconjunto T_1 de T que contém apenas os alvos que são cobertos por um sensor.

As linhas 4-16 são responsáveis por dar preferência aos alvos solitários. Já as linhas 6-11 constroem os subconjuntos T_{jl} de alvos que são cobertos por cada campo de visão de cada sensor. As linhas 13-15 encontram o campo de visão do sensor que cobre o maior número de alvos individuais e o orienta. Depois disso, o sensor cujo campo de visão foi orientado é retirado do conjunto de sensores C e os alvos que são cobertos por esse sensor nesse campo de visão são retirados de T.

Depois que todos os alvos solitários são cobertos, as linhas 17-21 são responsáveis por cobrir o restante dos alvos. É encontrado o campo de visão do sensor que cobre o maior número de

alvos (com o maior ranking) e esse é, então, orientado e retirado do conjunto C e os alvos que ele cobre são retirados de T. Esse processo continua até que um desses conjuntos seja vazio.

2.1.2. Heurística Híbrida Orientada aos Alvos (HTOH)

HTOH difere de GTOH apenas no cálculo de ranking dos campos de visão dos sensores e dos pesos dos alvos. Ao invés de atribuir peso 1 aos alvos que são cobertos por apenas um sensor e peso N (onde N representa o número de sensores) aos demais, cada alvo passa a ter peso igual a quantidade de sensores que o cobrem da seguinte forma:

$w_j = \sum_{i=1}^{N} \sum_{k=1}^{Q} a_{ik}^j a_{ik}^j \in A_{NxQ}^M$

Onde A_{NxQ}^M é uma matriz binária de cobertura composta de M alvos, N sensores e Q campos de visão dos sensores. E a_{ik}^j representa se um alvo m_j é coberto por um sensor n_i em um campo de visão q_k da seguinte forma:

$$\mathrm{a}_{ik}^j = \begin{cases} 1 & \textit{se } t_j \textit{ for coberto por } n_i \textit{ em } q_k \\ 0 & \textit{caso contrário} \end{cases}$$

2.2. Descrição de duas heurísticas para o MCMS baseadas em metaheurísticas

Nessa subseção serão apresentadas duas heurísticas baseadas em metaheurísticas que serviram como base para a realização deste trabalho. A primeira heurística é baseada na metaheurística GRASP que foi desenvolvida em [Resende, 1998] e a segunda heurística é baseada na metaheurística ILS que foi desenvolvida em [Salari, 2014].

2.2.1. Uma heurística baseada em GRASP para o MCMS (GRASP-Aleatório)

Como descrito em [Resende, 1998], GRASP (Greedy Randomized Adaptative Search Procedure) é uma metaheurística que vem sendo aplicada a uma grande gama de problemas de otimização combinatória. Cada iteração do GRASP consiste em duas fases, uma fase construtiva e uma fase de busca local.

Na fase construtiva, uma possível solução para o problema em questão é construída de forma iterativa, um elemento por vez. Em cada iteração, a escolha do próximo elemento a ser selecionado é determinada através da ordenação de todos os elementos em uma lista restrita de candidatos (LRC) que respeita uma função gulosa. Essa função mensura o benefício ao se selecionar cada elemento. A heurística é adaptativa porque os benefícios associados a todos os elementos são atualizados a cada iteração da fase construtiva e refletem as mudanças trazidas quando o elemento anterior foi selecionado. O componente probabilístico do GRASP é caracterizado através da seleção aleatória de um dos melhores candidatos de LRC, mas não necessariamente o melhor candidato. Essa técnica de seleção permite que soluções diferentes sejam obtidas a cada iteração do GRASP sem necessariamente comprometer o poder do componente guloso adaptativo do método.

Como a solução gerada pelo GRASP não é necessariamente um ótimo local de uma vizinhança pré-definida, é benéfico aplicar uma busca local com o intuito de melhorar cada solução construída.

Algorithm 2 GRASP genérico

procedure grasp($\alpha, MaxIter, RandomSeed$)
1: $BestSolutionFound = \emptyset$
2: **for** $k = 1, ..., MaxIter$ **do**
3: ConstructGreedyRandomizedSoln($\alpha, RandomSeed, p, J^*$);
4: LocalSearch(J^*);
5: **if** $w(J^*) > w(BestSolutionFound)$ **then**
6: $BestSolutionFound = J^*$;
7: **end if**
8: **end for**
9: **return** ($BestSolutionFound$)
end grasp;

O algoritmo acima mostra o pseudocódigo para o procedimento GRASP. O programa recebe como entrada um valor $0 \leq \alpha \leq 1$, gerado de forma aleatória a cada iteração, que promove um equilíbrio entre os critérios aleatórios e gulosos do procedimento, um número máximo de iterações e uma solução gerada de forma aleatória para o MCMS. A melhor solução encontrada até o momento ($BestSolutionFound$) é inicializada na linha 1. As iterações do procedimento acontecem entre as linhas 2 e 8. Cada iteração possui uma fase construtiva (linha 3) e uma fase de busca local (linha 4). Se necessário, a solução é atualizada na linha 5. GRASP retorna a melhor solução encontrada.

Algorithm 3 GRASP - Fase construtiva

procedure ConstructGreedyRandomizedSoln($\alpha, RandomSeed, J^*$)

1: $J^* = \emptyset$;
2: LRC = MakeLRC(α,J^*);
3: s = SelectRandomSolution(LRC);
4: J^* = getNeighbors(s);

end ConstructGreedyRandomizedSoln;

O pseudocódigo acima descreve a fase construtiva do GRASP que é responsável por gerar a LRC (linha 2) da seguinte forma:

$$LRC = \{e \in O | c(e) \leq min(c(e)) + \alpha(max(c(e)) - min(c(e)))\}$$

O maior custo $max(c(e))$ e o menor custo $min(c(e))$ dentre todas as soluções contidas em J^* são usados para definir um custo base c_{base} para a criação da LRC. Apenas as soluções contidas em J^* cujos custos são superiores a c_{base} entram na LCR. Então, um elemento contido em LRC é selecionado de forma aleatória na linha 3, e seus vizinhos são gerados na linha 4.

Para que a busca local fosse efetuada, a função de vizinhança f_1 que foi utilizada permite selecionar, dada uma solução aleatória $Z = \langle c_1,p_{n_1}\rangle, \langle c_2,p_{n_2}\rangle, ...\langle c_m,p_{n_m}\rangle$ para o MCMS, soluções que tem até 1 orientação diferente das contidas em Z variando apenas uma orientação de cada sensor.

2.2.2. Uma heurística baseada em ILS para o MCMS (ILS-Aleatório)

Como definido em [Salari, 2014], ILS (Iterated Local Search) é uma metaheurística que consiste, majoritariamente, em três partes: inicialização, busca local e perturbação. O algoritmo começa construindo uma solução possível para o problema de forma aleatória e tenta melhorá-la através de buscas locais em uma vizinhança pré-definida. Entretanto, para tentar escapar de um ótimo local, um procedimento de perturbação é desenvolvido e aplicado. O algoritmo é executado até não conseguir melhorar a qualidade da solução em um número sucessivo de iterações, Max_{iter}.

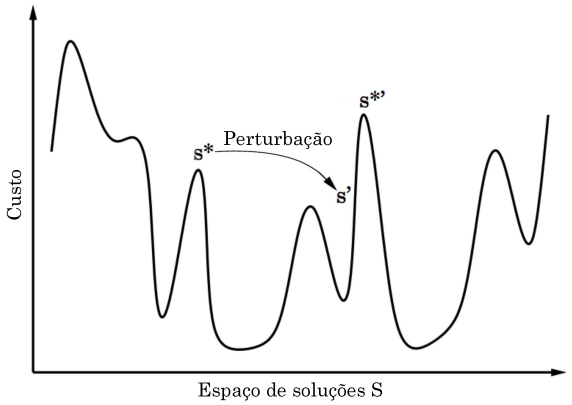

Figura 1: Representação de uma ILS. Começando de um máximo local s^*, uma perturbação é aplicada levando à solução s'. Depois de aplicar uma busca local em s' encontra-se um novo máximo local $s^{*'}$ que pode ser melhor que s^*.

Algorithm 4 ILS genérico

```
1:  CurrentSolution := Initialization();
2:  BestSolution := CurrentSolution;
3:  Iter = 0;
4:  repeat
5:      CurrentSolution := localSearch(CurrentSolution);
6:      if Cost(CurrentSolution) > Cost(BestSolution) then
7:          BestSolution := CurrentSolution;
8:          Iter = 0;
9:      else
10:         Iter = Iter + 1;
11:     end if
12:     CurrentSolution := Perturbation(CurrentSolution);
13: until Iter ≤ Max_iter
```

O algoritmo acima apresenta o pseudocódigo para o procedimento descrito. A linha 1 é responsável por gerar uma solução de forma aleatória ($CurrentSolution$). A linha 2 inicializa a variável $BestSolution$ com o mesmo valor de $CurrentSolution$. O laço presente entre as linhas 4 e 13 executará até que o critério de parada do algoritmo, ou seja, o número máximo de iterações, seja alcançado. A linha 5 é responsável por fazer a busca local a partir de $CurrentSolution$ e a condição presente entre as linhas 6 e 11 atualiza o valor de $BestSolution$ apenas se seu custo for pior que o custo de $CurrentSolution$. Caso contrário, o número de iterações $Iter$ é incrementado. Após esse processo, uma perturbação é realizada na solução armazenada em $CurrentSolution$ a fim de fugir de ótimos locais e recomeçar a partir de outra vizinhança.

Para que a perturbação seja feita de forma eficiente, um número k de mudanças é feito em $CurrentSolution$ até que, ao aplicar uma busca local em $CurrentSolution$ após sofrer a perturbação, o algoritmo não chegue no mesmo ótimo local.

Para que a busca local fosse efetuada, a mesma função de vizinhança f_1 definida anteriormente foi utilizada nesse algoritmo.

3. Metodologia

Nesta seção serão abordadas as modificações que foram propostas para cada uma das heurísticas apresentadas na seção anterior que visam resolver o problema de cobertura máxima de

alvos (MCMS).

3.1. Uma modificação da Heurística Híbrida Orientada aos Alvos (HTOHM)

Na Heurística HTOH Modificada utilizou-se HTOH como base, porém o método de escolha do ranking foi alterado. Em oposição à versão original, na qual escolhe-se o campo de visão do sensor com o maior ranking, a primeira e a segunda melhores opções são avaliadas. A escolha do ranking é feita de forma probabilística, tendo uma probabilidade de $33{,}33\%$ do segundo melhor ranking ser escolhido e $66{,}67\%$ do primeiro melhor ser escolhido. Assim, evita-se a obtenção de um resultado ótimo pela heurística.

Além disso foram feitas modificações no cálculo do Ranking. O mesmo g é feito fazendo o somatório de todos os pesos em cada campo de visão de cada sensor, ao invés do somatório de $1/w_i$, de acordo com a fórmula abaixo:

$R_{ik} = \sum_{j=1}^{M} \{w_i : t_j \in F(\langle s_i,p_k \rangle)\}\, i \in [1,N]; k \in [1,Q];$

3.2. Uma modificação da Heurística Gulosa Orientada aos Alvos (GTOHM)

GTOHM, assim como GTOH, dá prioridade aos alvos solitário, isto é, aos alvos que são cobertos apenas por um sensor. Logo após todos os alvos solitários serem cobertos, GTOHM começa a orientar os sensores para cobrir os alvos restantes levando em consideração o ranking dos campos de visão de cada sensor. Quanto mais alto o ranking, mais cedo o sensor será orientado e inserido no conjunto solução Z.

Um problema que foi observado no cálculo do ranking dos campos de visão utilizado em GTOH é que o mesmo permite falsos empates que podem dar prioridade erroneamente a certos campos de visão dos sensores. Para que isso fique mais claro, considere o cenário em que são fornecidos os 5 campos de visão de sensores com os rankings obtidos a partir do cálculo abaixo:

$R_1 = \frac{1}{4} + \frac{1}{2} + \frac{1}{4} + \frac{1}{2} = 1.5$
$R_2 = \frac{1}{6} + \frac{1}{4} + \frac{1}{5} = 0.616$
$R_3 = \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{3} = 1.83$
$R_4 = \frac{1}{4} + \frac{1}{4} = 0.5$
$R_5 = \frac{1}{2} = 0.5$

É fácil observar que R_4 e R_5 possuem rankings de mesmo valor, porém dizem respeito a cenários completamente diferentes. Enquanto o campo de visão do sensor cujo ranking é R_4 é capaz de cobrir dois alvos que são cobertos por 4 sensores cada, o campo de visão do sensor cujo ranking é R_5 é capaz de cobrir apenas um alvo que é coberto por apenas 2 sensores.

O cálculo acima garante que GTOH dê prioridade aos campos de visão de sensores que tem ranking maior, quando, na verdade, deveria dar prioridade aos campos de visão que conseguem cobrir mais alvos que por si só são cobertos por um número não muito grande de sensores. Levando em consideração o cenário descrito acima, GTOH poderia dar prioridade ao campo de visão cujo ranking é R_4 ao invés daquele cujo rank é R_5 (o cenário ideal).

Para que problemas como esse fossem evitados, sugere-se uma modificação no cálculo do rank que envolve multiplicá-lo pela fração $\frac{1}{media}$ (onde $media$ representa a média dos valores correspondentes à quantidade de sensores que cobrem cada alvo). Logo tem-se a seguinte alteração no cálculo do ranking de campos de visão de sensores:

$R_{ik} = (\sum_{j=1}^{M} \left\{ \frac{1}{w_i} : t_j \in F(\langle s_i,p_k \rangle) \right\}) * \frac{1}{media(w_{j1},w_{j2},...,w_{jm})}\, i \in [1,N]; k \in [1,Q];$

Após aplicar essa modificação no cenário encontrado acima, pode-se perceber que a modificação no cálculo do ranking permitirá que GTOH dê prioridade à R_4 ao invés de R_5, evitando, dessa forma, falsos empates. É importante destacar que tal alteração não afeta a ordem de prioridade ideal que o algoritmo deve respeitar ($R_3 > R_1 > R_5 > R_4 > R_2$).

3.3. Uma modificação da heurística baseada em GRASP para o MCMS (GRASP-GTOHM)

A heurística descrita nesta seção, assim como GRASP-Aleatório, também faz uso da técnica GRASP em sua construção. Porém, duas modificações foram propostas e implementadas. Como descrito anteriormente, a heurística (GRASP-Aleatório) recebe como parâmetro uma

solução que é gerada de forma aleatória para o MCMS. Portanto, a primeira alteração consiste em passar, como parâmetro de entrada, a solução obtida em GTOHM. Tal alteração tem como justificativa o fato de que GTOHM, dentre todas as quatro heurísticas propostas que não se baseiam em metaheurísticas (GTOH, GTOHM, HTOH e HTOHM), apresentou melhores resultados para o problema em questão.

A segunda modificação foi realizada na função de vizinhança f_1 que, como definido anteriormente, permite selecionar, dada uma solução aleatória $Z = \langle c_1,p_{n_1}\rangle, \langle c_2,p_{n_2}\rangle, ...\langle c_m,p_{n_m}\rangle$ para o MCMS, soluções que tem até 1 orientação diferente das contidas em Z variando apenas uma orientação de cada sensor. f_2 permite selecionar, dada a mesma solução aleatória Z, todas as soluções que tem até 1 orientação diferente das contidas em Z variando todas as orientações de cada sensor (uma por vez). Apesar dessa modificação resultar em um espaço de busca maior, muitos vizinhos que são deixados de lado em f_1 e que podem ter custos melhores foram incluídos na vizinhança gerada por f_2.

A essa heurística que foi apresentada nessa seção deu-se o nome GRASP(GTOHM).

3.4. Uma modificação da heurística baseada em ILS para o MCMS (ILS-GTOHM)

A heurística descrita nesta seção também faz uso da técnica ILS em sua construção. E duas modificações foram propostas e implementadas.

Como descrito anteriormente, a heurística baseada em ILS (ILS-Aleatório) recebe como parâmetro uma solução que é gerada de forma aleatória para o MCMS. Portanto, a primeira alteração consiste em passar, como parâmetro de entrada, a solução obtida em GTOHM pelo mesmo motivo justificado acima.

A segunda modificação foi realizada na função de vizinhança f_1 da mesma forma e pela mesma justificativa descrita acima.

A essa heurística que foi apresentada nessa seção deu-se o nome ILS(GTOHM).

4. Experimentos Computacionais

Para a realização dos experimentos utilizou-se um microcomputador com processador Intel Core i7-4510U 2.60GHZ com 8 GB de Memória RAM. Os dados foram gerados aleatoriamente através de um script desenvolvido em Python 2.7. Nele, inicialmente é recebido como entrada a quantidade de sensores e alvos a serem gerados. Por meio do módulo *random* são geradas as coordenadas dos sensores em um plano com dimensões $1000x1000$. Depois desse processo, os alvos são gerados seguindo o critério de estarem no raio de cobertura de algum dos sensores. O raio de cobertura dos sensores foi fixado em 100. Os scripts para as oito heurísticas aqui propostas foram escritos em Java e os gráficos foram gerados utilizando a biblioteca para R *ggplot2*.

Primeiramente, foram realizados testes fixando a quantidade de alvos em 150 e variando a quantidade de sensores de 150 até 250, com um espaçamento de 25, e, posteriormente, com a quantidade de sensores fixada e a quantidade de alvos variando da mesma forma. Para cada combinação de quantidade de alvo e sensor foram realizados testes com 10 bases de dados diferentes e para cada base de dados foram realizados 5 testes com cada algoritmo, podendo, assim, avaliar os diferentes tempos de execução e os resultados obtidos.

4.1. Avaliação de tempo de execução

Para avaliar o tempo de execução foram calculadas as médias de todas as experimentações para as quantidades de sensores e de alvos dadas. No gráfico que pode ser observado na Figura 2 foi fixada a quantidade de alvos em 150, com a quantidade de sensores variando em 25 unidades, e no gráfico observado na Figura 3 foram fixados os sensores com a quantidade de alvos variando da mesma forma. As heurísticas que se baseiam em procedimentos metaheurísticos e que partiram da solução obtida pela heurística GTOHM foram as que apresentaram maior tempo de execução, seguidas pelas que partiram de soluções geradas de forma aleatória. Este resultado foi possível porque a função de vizinhança que foi utilizada nas duas últimas é muito menor do que a função de vizinhança utilizada nas duas primeiras, o que resulta em um espaço menor de busca, levando um

tempo menor de execução. As heurísticas GTOH, HTOH e suas respectivas modificações apresentaram os melhores tempos de execução devido ao fato de serem heurísticas construtivas.

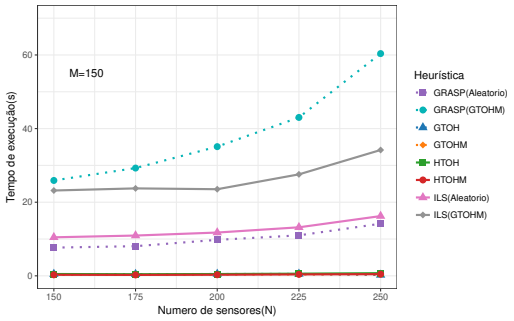

Figura 2: Análise de Tempo x Quantidade de sensores

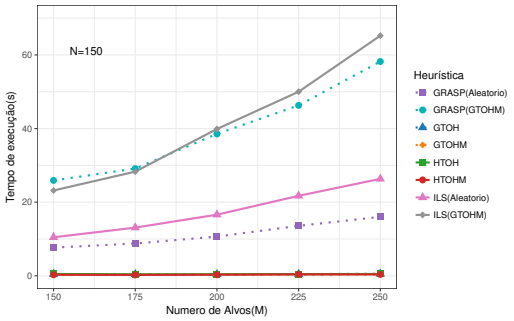

Figura 3: Análise de Tempo x Quantidade de alvos

4.2. Avaliação de qualidade

Na avaliação da qualidade foi dividida a quantidade de alvos cobertos por sensores pela quantidade total de alvos. Com essa razão se obteve um indicador de qualidade em que, quanto maior, melhor o desempenho da heurística. No gráfico observado na Figura 4 foi fixada a quantidade de alvos em 150, variando a quantidade de sensores. No gráfico observado na Figura 5 a quantidade de sensores foi fixada, variando a quantidade de alvos. Foi possível perceber que para todos conjuntos de dados o algoritmo GRASP(GTOHM) teve o melhor desempenho, seguido do VNS(GTOHM) e o GTOHM. Próximo deles também está HTOHM, VNS(Aleatório) e GRASP(Aleatório). Para todas as instâncias as heurísticas GTOH e HTOH tiveram desempenho bastante inferior quando comparadas às demais.

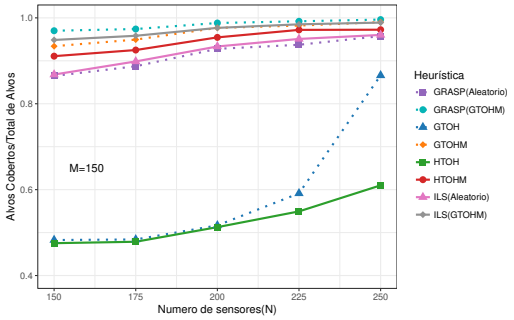

Figura 4: Análise de qualidade x Quantidade de sensores

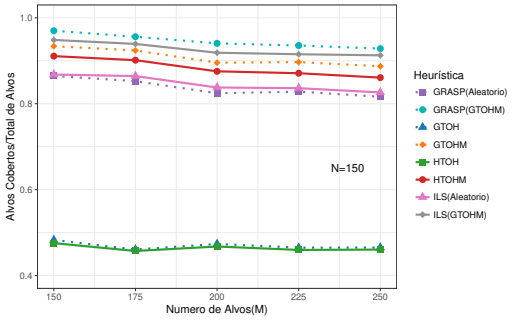

Figura 5: Análise de qualidade x Quantidade de alvos

4.3. Análise de dispersão entre tempo e qualidade

No gráfico de dispersão entre tempo e qualidade que pode ser observado na Figura 6 foi utilizada a média das experimentações para cada base de dados diferente. Nela é possível perceber que as heurísticas que se baseiam em metaheurísticas e que partiram das soluções obtidas pela heurística GTOHM possuem os melhores desempenhos, com poucos resultados inferiores a 90% dos alvos cobertos em relação ao total de alvos. Porém, é possível observar que as mesmas possuem os maiores tempos de execução.

As heurísticas GTOH e HTOH apesar de apresentarem baixos tempos de execução, apresentaram as piores qualidades das soluções. As heurísticas GTOHM e HTOHM mesmo possuindo tempos de execução inferiores a 1 segundo possuem desempenhos similares aos das heurísticas que se baseiam em metaheurísticas e que partem de soluções geradas de forma aleatória.

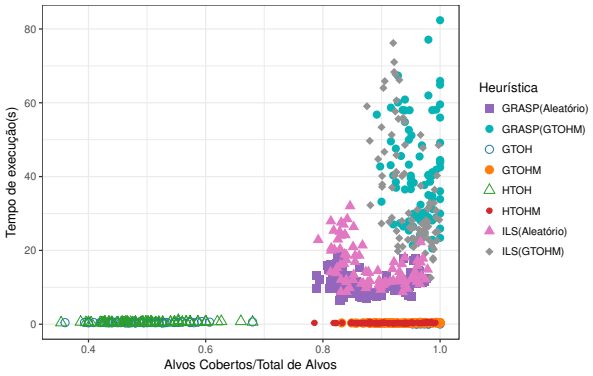

Figura 6: Análise de dispersão entre tempo e qualidade

4.4. Análise de robustez

Nos gráficos que podem ser observados nas figuras 7 e 8 tem-se um box plot que mostra a relação entre a qualidade das heurísticas com a quantidade de sensores e de alvos variando, respectivamente. Em ambos os gráficos é possível perceber que a qualidade das soluções aumenta de acordo com o aumento dos sensores, e diminuem de acordo como aumento dos alvos. Além disso, é possível observar que as heurísticas que se baseiam em metaheurísticas e que partem das soluções obtidas pela heurística GTOH possuem desempenhos superiores e que as heurísticas restantes possuíram desempenhos similares.

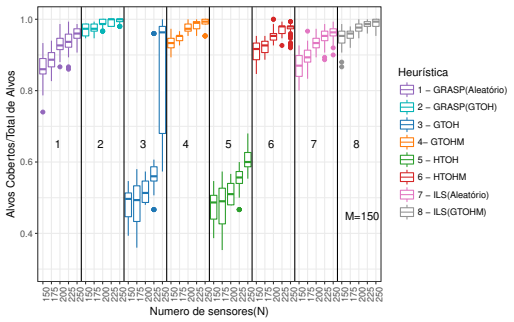

Figura 7: Análise de robustez x Quantidade de sensores

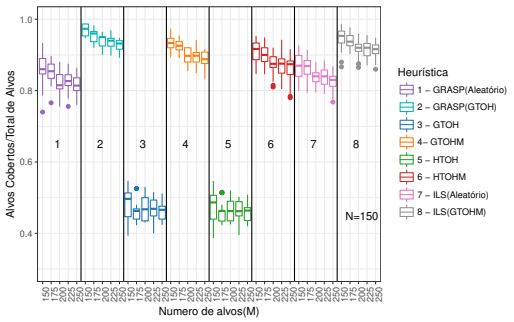

Figura 8: Análise de robustez x Quantidade de alvos

5. Conclusões

O presente trabalho apresentou modificações de quatro heurísticas existentes na literatura para a resolução do problema de cobertura máxima com mínimo número de sensores.

Quatro das heurísticas mostradas (GTOH, GTOHM, HTOH e HTOHM) são orientadas aos alvos, ou seja, dão prioridade aos alvos que são cobertos por apenas um sensor e posteriormente, com base em um sistema de ranking, procuram cobrir o restante dos alvos com o restante dos sensores. As quatro restantes (GRASP(Aleatório), GRASP(GTOHM), ILS(Aleatório) e ILS(GTOHM)) são baseadas em princípios metaheurísticos, sendo duas delas baseadas no procedimento GRASP e as outras duas na técnica ILS.

Para comparar empiricamente as oito heurísticas, uma bateria de experimentos foi planejada, executada e analisada. Foi observado, utilizando a mesma base de dados para testar todas elas, o tempo de execução e a qualidade da solução. Nos testes, variou-se o número de sensores e alvos em intervalos de 25 unidades.

Após a realização de todos os experimentos foi possível concluir que as modificações propostas nesse trabalho foram superiores em todos os testes, de acordo com as métricas adotadas,

sendo que GRASP(GTOHM) foi a que apresentou os melhores resultados. Isto foi possível devido a análises aprofundadas do cenário com objetivo de procurar métodos melhores para se chegar a soluções melhores.

Para trabalhos futuros, pode-se pensar em utilizar metaheurísticas que podem ser executadas a partir dos resultados obtidos nas duas heurísticas aqui propostas a fim de melhorar cada vez mais as soluções encontradas.

Referências

Ai, J. e Abouseid, A. (2006). Coverage by directional sensors in randomly deployed wireless sensor networks. *J. Comb. Optim.*, 11:21–41.

Alkyildiz, I., Melodia, T., e Chowdhury, K. (2007). A survey on wireless multimedia sensor networks. *Comput. Netw.*, 51:921–960.

Cai, Y., Lou, W., Li, M., e Li, M. (2007). Target-oriented scheduling in directional sensor networks. *IEEE INFOCOM*, p. 1550–1558.

Chakrabarty, K., Iyengar, S., Qi, H., e Cho, E. (2002). Grid coverage for surveillance and target location in distributed sensor networks. *Comput. IEEE Trans.*, 51:1448–1453.

Fusco, G. e Gupta, H. (2009). Selection and orientation of directional sensors for coverage maximization. *Proceedings of the 6th Annual IEEE Communications Society Conference on, IEEE Sensor, Mesh and Ad Hoc Communications and Networks*, SECON'09:1–9.

Hafsa, Z., Taslima, A., Mashura, T., e Ashikur, R. (2016). The coverage problema in visual sensor networks: A target oriented approach. *Journal of Network and Computer Applications*, 75:1–15.

Hörster, E. e Lienhart, R. (2006). On the optimal placement of multiple visual sensors. *Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks*, p. 111–120.

Munishwar, V. e Abu-Ghazaleh, N. (2011). Target-oriented coverage maximization in visual sensor networks. *Proceedings of the 9th ACM International Symposium on Mobility Management and Wireless Access*, p. 175–178.

Munishwar, V. e Abu-Ghazaleh, N. (2013). Coverage algorithms for visual sensor networks 9 (4), 45. *ACM Trans. Sens. Netw.*, 9:45.

Rashid, B. e Rehmani, M. (2016). Applications of wireless sensor networks for urban areas: a survey. *J. Netw. Comput. Appl.*, 60:192–219.

Resende, M. (1998). Computing approximate solutions of the maximum covering problem with grasp. *Journal of Heuristics*, 4:161–177.

Salari, M. (2014). An iterated local search for the budget constrained generalized maximal covering location problem. *Journal of Mathematical Modelling and Algorithms in Operations Research*, 13:301–313.

Soro, S. e Heinzelman, W. (2009). A survey of visual sensor networks. *Adv. Multimed.*