Universidade do Estado do Rio de Janeiro
Centro de Tecnologia e Ciências
Faculdade de Engenharia

Nicolás Bulla Cruz

Agrupamento espacial em robótica de enxame

Rio de Janeiro
2014

Nicolás Bulla Cruz

Agrupamento espacial em robótica de enxame

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Orientadora: Prof.ª Dr.ª Nadia Nedjah
Coorientadora: Prof.ª Dr.ª Luiza de Macedo Mourelle

Rio de Janeiro
2014

CATALOGAÇÃO NA FONTE

UERJ / REDE SIRIUS / BIBLIOTECA CTC/B

C957 Bulla Cruz, Nicolás.
Agrupamento espacial em robótica de enxame / Nicolás Bulla Cruz. 2014.
122 f.

Orientadora: Nadia Nedjah.
Coorientadora: Luiza de Macedo Mourelle.
Dissertação (Mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1. Engenharia Eletrônica. 2. Robótica – Dissertações. I. Nedjah, Nadia. II. Mourelle, Luiza de Macedo. III. Universidade do Estado do Rio de Janeiro. IV. Título.

CDU 007.52

Assinatura

Data

Nicolás Bulla Cruz

Agrupamento espacial em robótica de enxame

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Aprovado em 15 de agosto de 2014.

Banca Examinadora:

Prof.^{a} Dr.^{a} Nadia Nedjah (Orientadora)
Faculdade de Engenharia - UERJ

Prof.^{a} Dr.^{a} Luiza de Macedo Mourelle (Coorientadora)
Faculdade de Engenharia - UERJ

Prof. Dr. Luérbio Faria
Instituto de Matemática e Estatística - UERJ

Prof.^{a} Dr.^{a} Mariá Cristina Vasconcelos Nascimento Rosset
Universidade Federal de São Paulo - UNIFESP

Rio de Janeiro
2014

AGRADECIMENTOS

Dedico este trabalho a meus pais Gabino Bulla e Solange Cruz por acreditar em mim e apoiar sempre minhas decisões. A meu irmão Vladimir Bulla por incentivar e apoiar meu estudo de mestrado. A minha esposa Alejandra Bonilla por haver sido muito paciente durante nossa separação. A minha irmã Carolina Bulla que sempre estará presente em minhas conquistas.

Agradeço a minha orientadora Nadia Nedjah por sua paciência, compreensão e animo, além da disponibilidade e valiosas sugestões para realizar este trabalho. Agradeço à professora Luiza de Macedo Mourelle pelas aulas e pelas conversas para contextualizar-me no Brasil. Aos professores do PEL-UERJ pelas lições aprendidas nestes dois anos de curso.

Agradeço aos colegas de mestrado Alan, Alexandre, Carla, Fábio, Felipe, Heloisa, Kleber, Luneque, Leandro, Paulo, Rodrigo, Rafael, Rachel e Rogerio. Obrigado pelas conversas relacionadas com o projeto, com a língua portuguesa e com o Brasil. Os momentos de descontração no boliche, no kart, nos rodízios e churrascos. “Muchas gracias mis amigos”.

Agradeço a CAPES pelo apoio econômico brindado durante todo o curso de mestrado, sem o qual não poderia ter desenvolvido este trabalho em tempo integral.

"... Por que será que, os robôs quando armazenados num local vazio, eles se agrupam ao invés de ficarem sós? ..."

Dr. Alfred Lanning (Filme Eu, robô, baseado na obra de Isaac Asimov).

RESUMO

BULLA Cruz, Nicolás. Agrupamento espacial em robótica de enxame. 2014. 122 f. Dissertação (Mestrado em Engenharia Eletrônica) -- Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2014.

Os Sistemas Multi-Robôs proporcionam vantagens sobre um robô individual, quando da realização de uma tarefa com maiores velocidade, precisão e tolerância a falhas. Os estudos dos comportamentos sociais na natureza têm permitido desenvolver algoritmos bio-inspirados úteis na área da robótica de enxame. Seguindo instruções simples e repetitivas, grupos de robôs, fisicamente limitados, conseguem solucionar problemas complexos. Quando existem duas ou mais tarefas a serem realizadas e o conjunto de robôs é heterogêneo, é possível agrupá-los de acordo com as funcionalidades neles disponíveis. No caso em que o conjunto de robôs é homogêneo, o agrupamento pode ser realizado considerando a posição relativa do robô em relação a uma tarefa ou acrescentando alguma característica distintiva. Nesta dissertação, é proposta uma técnica de clusterização espacial baseada simplesmente na comunicação local de robôs. Por meio de troca de mensagens entre os robôs vizinhos, esta técnica permite formar grupos de robôs espacialmente próximos sem precisar movimentar os robôs. Baseando-se nos métodos de clusterização de fichas, a técnica proposta emprega a noção de fichas virtuais, que são chamadas de *cargas*, sendo que uma carga pode ser estática ou dinâmica. Se uma carga é estática permite determinar a classe à qual um robô pertence. Dependendo da quantidade e do peso das cargas disponíveis no sistema, os robôs intercambiam informações até alcançar uma disposição homogênea de cargas. Quando as cargas se tornam estacionárias, é calculada uma densidade que permite guiar aquelas que estão ainda em movimento. Durante as experiências, foi observado visualmente que as cargas com maior peso acabam se agrupando primeiro enquanto aquelas com menor peso continuam se deslocando no enxame, até que estas cargas formem faixas de densidades diferenciadas para cada classe, alcançando assim o objetivo final que é a clusterização dos robôs.

Palavras-chave: Clusterização espacial. Robótica de enxame. Computação distribuída. Inteligência de enxame.

ABSTRACT

BULLA Cruz, Nicolás. *Spatial clustering in swarm robotics*. 2014. 122 f. Dissertação (Mestrado em Engenharia Eletrônica) -- Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2014.

Multi-Robots Systems provide advantages over a single robot when performing a task, achieving a greater speed, higher accuracy and better fault tolerance. The studies of social behavior in nature have allowed to develop bio-inspired algorithms useful in swarm robotics. Following simple and repetitive rules, groups of robots can provide solutions to complex problems. When two or more tasks to be executed by a set of heterogeneous robots, it is possible to cluster the robots according to their intrinsic features. When homogeneous robots are used, the clustering may be achieved by considering the robot relative position regarding the location where the task has to be performed or adding some other distinct feature. In this dissertation, a technique for spatial clustering simply based on local communication between robots is proposed. Through the message exchange between neighboring robots, this technique allows cluster formation without robot movement. Based on the token clustering methods, the proposed technique employs a virtual token, which is called a *load*. The load allows identifying the class to which a robot belongs. Depending on the amount and weight of the loads available in the system, the robots interchange information to achieve uniform load distribution. When the loads become stationaries, a density is calculated as to guide the remaining loads that are still in motion. As a consequence, the loads of higher weight cluster first and the those of lower weight continue shifting through the swarm, until they start forming different density ranges for each class, thereby achieving the final aim which is robot clustering.

Keywords: Spatial clustering. Swarm robotics. Distributed computing. Swarm intelligence.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

SUMÁRIO

INTRODUÇÃO

INTELIGÊNCIA de Enxame foi introduzida por Beni e Wang em 1989 (BENI; WANG, 1993). É um ramo da inteligência artificial dedicado ao comportamento coletivo de sistemas descentralizados e auto-organizáveis. Seguindo regras simples, agentes autônomos conseguem realizar tarefas complexas, interagindo entre eles e com o ambiente.

Estudos de comportamentos de animais na natureza têm permitido o desenvolvimento de técnicas computacionais que permitem resolver problemas complexos. Algumas destas técnicas bio-inspiradas são baseadas no comportamento coletivo de insetos sociais. As tarefas que não podem ser realizadas por um único indivíduo, podem ser efetuadas por um grupo de indivíduos. A partir destes estudos foram desenvolvidos vários algoritmos entre eles a otimização por colônia de formigas (DORIGO; MANIEZZO; COLORNI, 1996) e colônia artificial de abelhas (KARABOGA; BASTURK, 2007), entre outros. Os algoritmos desenvolvidos têm sido amplamente aplicados na área dos sistemas distribuídos, obtendo resultados satisfatórios na solução de problemas de alta complexidade.

Os sistemas multi-robôs proporcionam vantagens sobre um robô individual, quando da realização de uma tarefa proporcionando maiores velocidades, precisão e tolerância a falhas (MARJOVI; CHOOBDAR; MARQUES, 2012). O desenvolvimento de robôs econômicos, projetados para estudos na robótica de enxame, permitiram a implementação dos algoritmos de inteligência enxame em robôs reais. Em um sistema multi-robô, grupos de robôs, fisicamente limitados, conseguem solucionar problemas complexos.

Para aproveitar melhor o trabalho cooperativo de grandes grupos de robôs, é necessária a coordenação das tarefas a serem realizadas. Dividindo as tarefas em sub-tarefas, é possível concluir todas as atividades usando menor tempo e obtendo melhores resultados. A divisão das tarefas requer uma divisão do enxame em subgrupos de robôs. Nesta dissertação cada um dos grupos será identificado com uma classe, *i.e.* quando um robô pertence a um grupo **A**, será dito que o robô pertence à classe **A**. A alocação de tarefas, na robótica, tem sido estudada com a perspectiva da tarefa e a sua complexidade (MEN-

DONÇA; NEDJAH; MOURELLE, 2014), mas pode também ser abordada com a perspectiva da posição relativa aos robôs.

A clusterização engloba um conjunto de técnicas computacionais que permitem de forma automática agrupar elementos que apresentam certo grau de semelhança. Estas técnicas de agrupamento são principalmente utilizadas na mineração de dados, onde grandes quantidades de dados precisam ser classificados para depois serem analisados com o objetivo de gerar conhecimentos. Os conceitos básicos das técnicas de clusterização podem ser ampliados e combinados para obter resultados em outros campos de pesquisa, tais como: reconhecimento de padrões, processamento de imagens, entre outros.

O objetivo desta dissertação é propor e implementar um algoritmo distribuído de clusterização espacial para robótica de enxame, generalizada para mais do que 2 classes. Sem conhecimentos globais do sistema nem a sua posição dentro do enxame, os robôs de um enxame deveriam conseguir trabalhar em conjunto para resolver o problema de clusterização.

A principal contribuição deste projeto é a generalização da ideia de clusterização em 2 classes apresentada em (DI CARO; DUCATELLE; GAMBARDELLA, 2012), para poder aplicá-la na clusterização em um número qualquer de classes pré-definido. Utilizando como estratégia a movimentação de fichas virtuais, os robôs compartilham informação de maneira local, concentrando as fichas em certas localizações do espaço ocupado pelo enxame. Na mesma forma em que as formigas concentram e organizam os corpos (BONABEAU; DORIGO; THERAULAZ, 1999), o algoritmo utiliza a informação da concentração das fichas para inferir uma densidade, que pode guiar o movimento das fichas virtuais espalhadas entre os robôs do enxame. Também, analisar o comportamento do algoritmo em diferentes configurações de enxame. Para isto, o algoritmo é avaliado variando o tamanho de enxame, o número de classes e com proporções diferentes das classes a serem formadas. Os primeiros resultados obtidos durante este trabalho foram publicados em (BULLA; NEDJAH; MOURELLE, 2014a) e (BULLA; NEDJAH; MOURELLE, 2014b).

Os resultados obtidos das experiências são satisfatórios, conseguindo a clusterização de até 4 classes, utilizando enxames de até 28 robôs. O tempo utilizado pelos robôs para clusterizar é aceitável em todas as experiências realizadas.

O restante desta dissertação está organizado em cinco capítulos, cujos conteúdos são resumidamente descritos a seguir.

Inicialmente, o Capítulo 1 apresenta uma introdução teórica ao problema de clusterização. Também, são descritos alguns métodos de agrupamento geralmente utilizados na área de mineração de dados. Por último, é abordada a clusterização na área de robótica.

O Capítulo 2 apresenta de maneira sucinta os trabalhos relacionados à clusterização na robótica de enxame e redes *ad hoc*. São apresentadas diferentes abordagens da clusterização em sistemas distribuídos.

O Capítulo 3 descreve detalhadamente o algoritmo de clusterização espacial idealizado para 2 classes, assim como a sua generalização para qualquer número de classes e utilizado no desenvolvimento deste trabalho. São descritos os termos de *carga* e *densidade de cargas estáticas*, sendo estes os aspectos básicos do algoritmo de clusterização proposto.

O Capítulo 4 apresenta a implementação do algoritmo proposto utilizando robôs do tipo *Kilobot* (RUBENSTEIN; AHLER; NAGPAL, 2012). São descritas as características do processo de comunicação utilizado pelos robôs. Em seguida, são analisados os resultados obtidos da execução do algoritmo nos enxames de robôs. Para isso, são utilizados diferentes configurações de enxame para avaliar o tempo de convergência.

Finalmente, o Capítulo 5 completa esta dissertação, apresentando as principais conclusões obtidas do desenvolvimento do presente trabalho. São apresentadas também possíveis melhorias a partir do algoritmo proposto e as direções para futuros trabalhos.

Capítulo 1

CLUSTERIZAÇÃO

MÉTODOS de clusterização ou agrupamento é a denominação dada para o conjunto de técnicas computacionais cujo propósito consiste em classificar diferentes elementos, entidades ou registros em grupos, dependendo de características ou graus de semelhança que estes possuam. Os grupos formados, chamados de *clusters*, são o resultado do procedimento classificativo aplicado nos elementos a serem agrupados. Geralmente, os elementos agrupados em um mesmo *cluster* se concentram em uma região densa e possuem características particulares próximas entre si. A ideia básica da clusterização consiste em colocar em um mesmo grupo, os indivíduos que são similares de acordo com algum critério pré-determinado, (LINDEN, 2009). As técnicas de clusterização baseiam-se normalmente em uma função de *dissimilaridade*, que determina a *distância* entre dois elementos considerando uma ou mais características pré-definidas. Os grupos determinados em um sistema devem ser descritos em termos de *homogeneidade interna* e *separação externa*. Em outros termos, os elementos de um mesmo grupo devem ser mutuamente similares, e também diferentes dos elementos de outros grupos.

A clusterização é uma ferramenta útil para reduzir a dimensão de um conjunto de dados, reduzindo uma ampla gama de objetos em uma informação que diz respeito ao centro do seu conjunto. Em (JAIN; DUBES, 1988), é descrita a clusterização como um método que utiliza o aprendizado não supervisionado ou auto-organizável, ou seja, não existe um especialista que determine como deve ser formado cada grupo ou o que cada padrão representa.

Consequentemente, as técnicas de agrupamento têm sido aplicadas em diferentes áreas de conhecimento. Em *mineração de dados*, a clusterização tem sido utilizada principalmente para a identificação de distribuições em um determinado conjunto de dados, per-

mitindo apreciar as propriedades de cada *cluster*. Também, permitem descobrir padrões gerais das distribuições ocultas considerando os diferentes atributos dos dados (LAURO, 2008).

Atualmente, existem diversos algoritmos heurísticos de clusterização de dados (XU; WUNSCH, 2009). Os resultados da clusterização não são preditivos, e a formação dos *clusters* está relacionada às regras estabelecidas. Quando existem diferentes critérios a serem levados em conta, pode acontecer que um elemento possa ser agrupado em 2 ou mais grupos distintos. Embora a clusterização convirja em uma solução, não é possível estabelecer qual dos critérios é o melhor.

1.1 Conceitos Básicos de Clusterização

Nesta seção, serão apresentados alguns conceitos básicos, frequentemente encontrados na área da clusterização, e que serão utilizados no decorrer deste trabalho.

1.1.1 Elemento e atributo

Na área de análise de agrupamento são frequentemente utilizados os termos *elemento*, *objeto*, *entidade*, *item*, *observação*, *membro*, para denotar um item individual de um conjunto de dados. Neste trabalho serão utilizados principalmente os termos *membro* e *elemento*. Já os atributos são as características de cada um dos elementos do sistema. No trabalho desenvolvido e descrito nesta dissertação, a densidade de carga e sua classe são os dois atributos levados em conta pelo processo de clusterização, conforme será detalhado no Capítulo 3.

1.1.2 Cluster

Geralmente nas técnicas de clusterização, o termo *cluster* está associado com os grupos, subconjuntos ou categorias em que os elementos ou objetos são agrupados. Embora, esta definição não seja precisa, ela é universalmente aceita (XU; WUNSCH, 2009). Não obstante, existem algumas definições estabelecidas em (EVERITT, 1974) que são precisas e também aceitas:

- Um *cluster* é um conjunto de elementos que são similares, adicionalmente, os elementos de clusters diferentes não são similares.

- Um *cluster* é uma agregação de pontos no espaço de busca, tal que, a distância entre dois pontos quaisquer no cluster é menor do que a distância entre qualquer ponto no cluster e outro ponto fora dele.

- Um *cluster* pode ser descrito como uma região contínua num espaço de densidade de pontos relativamente alta, separada de outro *cluster* por regiões de densidade de pontos relativamente baixa.

1.1.3 Distância e similaridade

Alguns métodos de agrupamento determinam a similaridade entre dois objetos, definindo um conceito de distância entre eles (HAN; KAMBER; PEI, 2006). As distâncias podem ser definidas no espaço Euclidiano, em um espaço vetorial, ou qualquer outro espaço. Em outros métodos, a similaridade é definida pela conectividade baseada na densidade ou contiguidade, que pode não ser baseada na distância absoluta entre os dois objetos (HAN; KAMBER; PEI, 2006).

A distância e a similaridade representam fatores importantes na clusterização (LAURO, 2008). O grau de similaridade depende da medida escolhida. O conhecimento do problema facilita a escolha da medida a ser implementada. Existem diversas formas de calcular esta distância, sendo distância de *Minkowski* a generalização das métricas no espaço Euclidiano (LAURO, 2008), conforme Equação 1:

$$y(i,j) = \left(\sum_{k=1}^{d} |x_{ik} - x_{jk}|^q \right)^{\frac{1}{q}}, \tag{1}$$

onde $y(i,j)$ é a distância entre os elementos i e j, x_{ik} é o atributo k medido do elemento i, x_{jk} o atributo k medido do elemento j, d representa o número total de atributos ou características medidas, e q é a ênfases que se deseja dar às grandes distâncias, com $q \geq 1$. Os casos particulares de esta distância são quando $q = 2$, distância conhecida como *Euclidiana*, e $q = 1$, distância conhecida como *Manhattan*.

1.1.4 Matriz de similaridade

A matriz de similaridade ou dissimilaridade representa a tabela onde são relacionadas as medidas de distância entre os objetos envolvidos no processo de agrupamento (ARAÚJO, 2012). Em outros termos, cada elemento desta matriz corresponde à distância $y(i,j)$ entre os objetos i e j envolvidos no cálculo. Note que, tem-se $y(i,j) = y(j,i)$ e $y(i,i) = 0$. A

distância $y(i, j)$ é um inteiro não negativo que se aproxima de 0 quando os objetos são muito similares e cresce à medida que os mesmos apresentem uma maior diferença.

O algoritmo de clusterização espacial utilizado neste trabalho e apresentado no Capítulo 3 não utiliza a noção de matriz de similaridade, nem calcula explicitamente a distância entre os robôs. No entanto, no Capítulo 4 serão apresentados os valores respectivos da matriz de similaridade e da distância de maneira explícita com o propósito de evidenciar as propriedades de clusterização do algoritmo proposto.

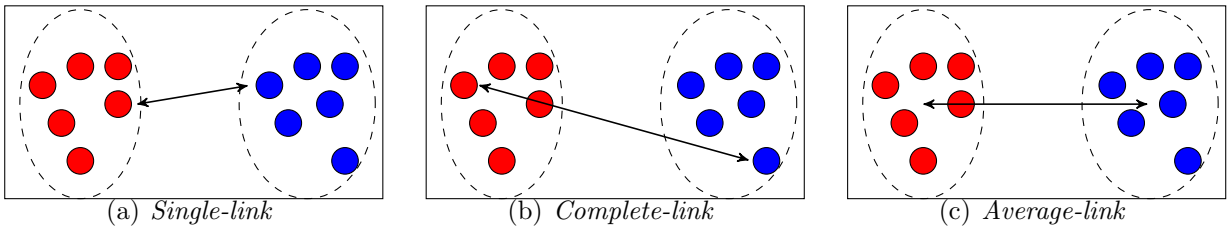
1.2 Algoritmos de Clusterização

A seleção do algoritmo de agrupamento depende da natureza dos dados tais como a medida de temperatura, densidade, ou qualitativos como a cor; do tipo de elementos a serem agrupados tais como robôs ou fichas; e do objetivo do agrupamento, por exemplo, agrupar livros por tamanho ou por matéria. Embora existam muitos algoritmos, não existe um algoritmo de agrupamento ideal que pode ser aplicado para todos os tipos de dados. Alguns métodos de agrupamento serão apresentados com o propósito de entender melhor os conceitos e comparar os métodos existentes com o método proposto neste trabalho.

1.2.1 Algoritmos Hierárquicos

Os algoritmos hierárquicos classificam iterativamente os elementos de um conjunto, considerando a distância que existe entre os elementos (LAURO, 2008). Um algoritmo hierárquico pode ser *aglomerativo* ou *divisivo*. Um algoritmo é dito aglomerativo quando cada elemento do conjunto começa formando um grupo separado, sendo que número inicial de grupos coincide com o número de elementos a serem agrupados. No decorrer do processo de clusterização, os grupos com distâncias próximas são combinados sucessivamente até que se atinja uma condição de parada tais como formar certo número de grupos pré-definido ou esperar até que todos os elementos conformem um único grupo. Em contraste, um algoritmo é dito divisivo quando todos os elementos começam formando um único grupo, e a cada iteração o grupo é dividido em grupos menores baseando-se na distância entre os elementos, até que uma determinada condição de parada seja atingida tais como formar certo número de grupos pré-definido ou esperar até que cada grupo contiver somente um elemento. Os critérios de proximidade entre dois *clusters* podem ser definido pela mínima distância (*single-link*), dada pela distância entre os elementos mais próximos,

a máxima distância (*complete-link*), dada pela distância entre os elementos mais distantes, ou distância meia (*average-link*), dada pela distância entre os centroides (HAN; KAMBER; PEI, 2006), na Figura 1 são exemplificados estes critérios de proximidade.



(a) *Single-link* (b) *Complete-link* (c) *Average-link*

Figura 1: Critérios de proximidade

O algoritmo denominado AGNES (*Agglomerative Nesting*) (HAN; KAMBER; PEI, 2006) é um algoritmo baseado no método hierárquico aglomerativo. Para realizar a junção de 2 grupos, o algoritmo utiliza um dos critérios de proximidade supracitados. Na Figura 2, é apresentado um exemplo da execução da técnica. Dados os elementos $\{a, b, c, d, e\}$, são criados grupos que contem um único elemento e é calculada a distância entre os grupos. A menor distância encontrada corresponde a distância entre os grupos que contem os elementos a e b, gerando o grupo $\{a, b\}$. Recalculando as distâncias entre os grupos de forma iterativa, são identificados novos grupos, até formar um único grupo constituído por todos os elementos. Se o critério de parada é, por exemplo, a obtenção de 2 grupos, a solução estaria no Passo 3, onde são identificados os conjuntos $\{a, b, c\}$ e $\{d, e\}$.

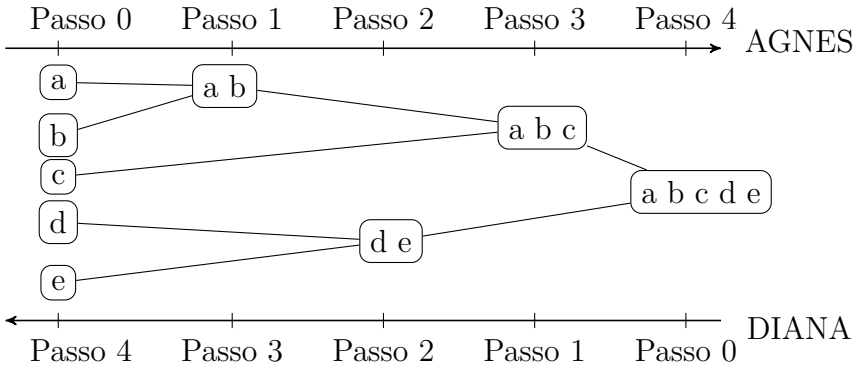


Figura 2: Exemplo da evolução dos algoritmos hierárquicos aglomerativos (AGNES) e divisivo (DIANA)

O algoritmo denominado DIANA (*Divisive Analysis*) (HAN; KAMBER; PEI, 2006) é baseado no método hierárquico divisivo. O agrupamento é dividido em dois a cada iteração, seguindo um dos critérios supracitados, até que seja atingido o número de grupos

desejado ou até que cada grupo seja formado de um único elemento. Este algoritmo pode ser visualizado como um algoritmo aposto do AGNES. Na Figura 2, é apresentado um exemplo da execução da técnica. Um único grupo formado por os elementos $\{a, b, c, d, e\}$ é dividido iterativamente. Ao final, cada um dos grupos é formado por um único elemento. Se o critério de parada é, por exemplo, a obtenção de 3 grupos, a solução estaria no Passo 2, onde são identificados os conjuntos $\{a, b\}$, $\{c\}$ e $\{d, e\}$.

1.2.2 Algoritmos de Particionamento

Nos algoritmos de particionamento, os elementos a serem clusterizados são agrupados em P partições (LAURO, 2008). Os elementos de cada partição ou grupo possuem características semelhantes. Cada grupo formado deve conter pelo menos um elemento, e cada elemento deve pertencer a um único grupo. Então, para n elementos deve se satisfazer que $P \leq n$. Definido o número de partições ou grupos, é gerada uma partição inicial aleatória, representando os centróides dos grupos. Iterativamente, é calculada a distância dos elementos aos centróides e recalculada a posição dos novos centróides. O critério de parada da técnica é geralmente determinado por um limiar de proximidade entre os elementos de um mesmo grupo, ou um limite de distanciamento entre grupos diferentes.

O algoritmo *K-médias* (JAIN; DUBES, 1988) é um dos métodos mais populares na área de *mineração de dados*. Este algoritmo visa particionar n elementos em P grupos. Cada elemento é agrupado levando em conta a proximidade com o centróide ou média do grupo. Inicialmente, são estabelecidas aleatoriamente as posições dos centróides. No decorrer do processo, é calculada a distância de cada elemento ao centróide, associando o elemento ao grupo cuja distância seja menor. Uma vez associados todos os elementos, é recalculada a posição dos centróides baseado na média dos elementos do grupo. Iterativamente, são reagrupados os elementos considerando os novos centróides é recalculada a posição dos centróides. O algoritmo converge quando a posição dos centróides permanece constante por certo número de iterações.

Na Figura 3 é apresentado um exemplo ilustrativo do movimento dos centróides para uma clusterização de 15 elementos em 3 classes. Os elementos estão localizados em um plano de duas dimensões e representados por círculos. Os centróides são representados por '$\times$', '$\triangle$' e '$+$'. As setas representam a movimentação dos centróides.

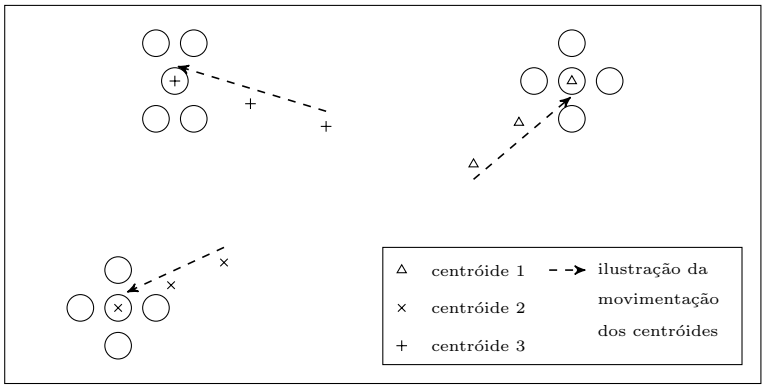
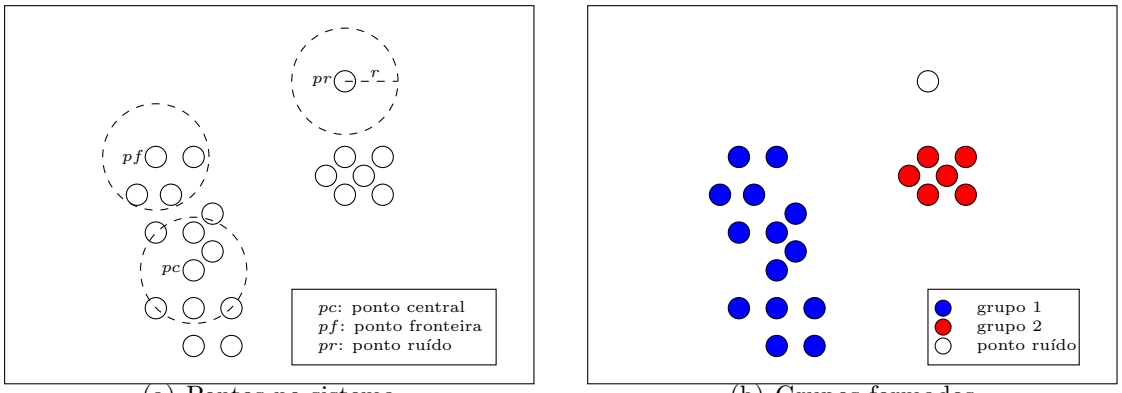


Figura 3: Exemplo ilustrativo da mudança da posição dos centróides no algoritmo *K-meios* com $P = 3$

1.2.3 Algoritmos Baseados em Densidade

A clusterização baseada na densidade visa agrupar elementos levando em consideração o número de elementos dentro de um raio de vizinhança. Os métodos baseados em densidade geram grupos aglomerados em regiões com alta densidade separadas por regiões de baixa densidade. Os elementos que formam as regiões de baixa densidade, ou seja, sem uma vizinhança, são considerados como ruído.

O DBSCAN (*Density Based Spatial Clustering of Applications with Noise*) é o primeiro algoritmo baseado em densidade (PASCUAL; PLA; SÁNCHEZ, 2007). O algoritmo DBSCAN introduz os conceitos de ponto *central*, ponto de *fronteira* e ponto de *ruído*. O algoritmo começa selecionando um ponto η arbitrário. Se este ponto apresenta na vizinhança uma quantidade de pontos maior ou igual do que um limiar pré-definido, então η é denominado *ponto central*. Caso contrário, *i.e.* se η não é um ponto central, então é selecionado um outro ponto que é tratado da mesma maneira, até percorrer todo o conjunto de dados. Quando η é um ponto central, é criado um grupo com os vizinhos *denso-alcançáveis* deste, ou seja, com os vizinhos próximos. Um grupo pode ter mais de um ponto central. Os pontos não centrais do grupo são denominados *pontos de fronteira*. Os pontos que não são centrais ou fronteiriços são denominados *pontos de ruído*. Na Figura 4 (a) é apresentado um exemplo ilustrativo do algoritmo DBSCAN para um raio de vizinhança r e um mínimo de pontos na vizinhança de 5. Os pontos pc, pf e pr representam respectivamente, um dos pontos centrais, um dos pontos fronteiriços e um dos pontos de ruído no conjunto de dados. A Figura 4 (b) mostra o resultado da clusterização.



(a) Pontos no sistema (b) Grupos formados

Figura 4: Exemplo ilustrativo do algoritmo DBSCAN

1.2.4 Algoritmos Baseados em Grafos

Baseando-se na *Teoria de Grafos*, foram desenvolvidos métodos de agrupamento. Um grafo é um modelo matemático que representa uma relação entre os objetos (LINDEN, 2009). Um grafo é definido por $G = (V, E)$, onde V é um conjunto finito de pontos, chamados de nós ou vértices e E é uma relação entre vértices. Os elementos de E são chamados de arestas. O grau ou a conectividade de um vértice é determinado pelo número de arestas que entram ou saem dele. Se as arestas de um grafo apresentar uma direção, o grafo é chamado de direcionado, caso contrário é chamado de não-direcionado. Um grafo é dito conectado se existe um caminho de arestas entre quaisquer dois vértices do grafo. Quando cada um dos vértices está relacionado por meio de uma aresta com todos os outros vértices, o grafo é dito completo. Um *grafo de clique* é um grafo em que cada componente conectado é um grafo completo.

Tendo em consideração estas definições, uma forma de agrupar elementos consiste em definir um grafo baseado nos seus dados e supor que o grafo obtido representa uma clique corrompida. Desta maneira é procurado o número mínimo de arestas a serem adicionadas ou retiradas de forma a obter uma clique. Os sub-grafos completos obtidos representarão os *clusters* mais adequados para os dados. Na Figura 5 é apresentado um exemplo ilustrativo da clusterização baseada em cliques. No grafo original as arestas marcadas com $\times$ serão removidas. No grafo de cliques obtidos as arestas pontilhadas foram inseridas.

(a) Grafo original (b) Cliques obtidos

Figura 5: Exemplo ilustrativo da clusterização baseada em grafos

1.3 Clusterização em Robótica

Os sistemas multi-robôs consistem principalmente de robôs simples que geralmente possuem baixa capacidade computacional devido às restrições do custo (MARJOVI; CHOOBDAR; MARQUES, 2012). Contudo, trabalhando conjuntamente os robôs conseguem resolver problemas complexos. Para obter uma melhor eficiência na solução de problemas, o problema principal é dividido em subproblemas que são, em seguida, distribuídos entre os robôs individuais ou grupos de robôs.

A clusterização na robótica de enxame tem dois enfoques dependendo dos elementos a serem agrupados: *(i)* a clusterização de fichas, ou elementos passivos, distribuídos em um ambiente; *(ii)* a clusterização dos próprios robôs.

Na clusterização de fichas, ou elementos passivos, é estudado o comportamento de enxames de robôs que têm a capacidade de movimentar fichas de um ponto a outro (BECKERS; HOLLAND; DENEUBOURG, 1994). Utilizando heurísticas bio-inspiradas em sistemas de inteligência coletiva, os robôs conseguem agrupar elementos homogêneos em um só *cluster* e, dependendo das características sensoriais dos robôs, conseguem agrupar elementos heterogêneos em diferentes *clusters*. Um dos aspectos a ser levado em conta, durante a clusterização de fichas, é a física referente à movimentação dos elementos passivos, assim como a física referente à movimentação dos próprios robôs.

A clusterização de robôs procura gerenciar a divisão de grupos grandes de robôs, com o propósito de acomodar as tarefas simples correspondentes à subproblemas identificados (MENDONÇA; NEDJAH; MOURELLE, 2014). Quando existem duas ou mais tarefas a serem realizadas e o conjunto de robôs é heterogêneo, é possível agrupar os robôs segundo as funcionalidades neles disponíveis. No caso em que os robôs a serem clusterizados são homogêneos, e não exista nenhuma característica adequada para determinar a forma que estes devam ser clusterizados, é preciso então agregar um critério que permita realizar a

distinção entre os grupos de robô. Para realizar uma clusterização balanceada de robôs homogêneos não existem algoritmos consolidados. Uma abordagem deste problema é realizar a clusterização tendo como critério a distância entre os robôs e os locais onde as tarefas devem ser realizadas. Outra abordagem é implementar o uso de *tokens*. Um *token* é uma carga virtual que permite relacionar um robô com uma classe, ou a ausência dela com outra classe diferente. Neste trabalho é utilizada a abordagem por cargas virtuais, explicado no Capítulo 3.

1.4 Considerações Finais do Capítulo

Neste capítulo foram apresentados conceitos básicos à clusterização. A clusterização são técnicas de agrupamento de dados que consideram os graus de semelhança entre os objetos. Também foram explicados de forma geral, alguns métodos de agrupamento amplamente utilizados na área de mineração de dados. Por último, foi abordada a clusterização na área da robótica. O capítulo seguinte apresenta um estudo sobre os principais trabalhos realizados com algoritmos de clusterização em robótica e redes sem-fio.

Capítulo 2

TRABALHOS RELACIONADOS

ESTE capítulo, organizado em duas seções, apresenta alguns trabalhos relacionados à clusterização com aplicação em diferentes áreas. Na Seção 2.1 são apresentadas implementações de clusterização de elementos passivos por robôs assim como clusterização de robôs. Na Seção 2.2 são apresentados algoritmos de clusterização de redes de sensores sem fio, WSNs (*Wireless Sensor Networks*), e de redes *ad hoc* móveis ou *MANET* (*Mobile Ad Hoc Network*).

2.1 Clusterização na Robótica

Na área de robótica, os trabalhos relacionados à clusterização geralmente visam utilizar os robôs como agentes que executam o agrupamento, sendo que os elementos passivos são os objetos a serem agrupados. Por outro lado, estão os trabalhos que visam agrupar os próprios robôs. O objetivo principal desta abordagem consiste em dividir em grupos os robôs para a realização de tarefas ainda não estabelecidas, assim como a validação de teorias de Inteligência de Enxame inspirados em comportamentos de animais sociais.

2.1.1 Clusterização de Elementos Passivos

Em (BECKERS; HOLLAND; DENEUBOURG, 1994), é apresentado um dos primeiros trabalhos que apresenta um algoritmo de clusterização. Neste trabalho é implementado em robôs um comportamento bio-inspirado em insetos sociais. Para limpar seus formigueiros, algumas espécies de formiga juntam corpos e partes de corpos de formigas mortas em regiões específicas do formigueiro. O mecanismo básico deste processo é uma atração entre os elementos mortos mediada pelas formigas (BONABEAU; DORIGO; THERAULAZ, 1999). Pequenos amontoados se formam e vão crescendo atraindo uma maior quantidade

de corpos naquela região do espaço. Visando demonstrar que os comportamentos bio-inspirados representam uma vantagem em relação aos comportamentos que não são, foi implementado um mecanismo de coordenação indireto entre agentes, chamado de *estigmergia* (HAMANN, 2010) onde a comunicação acontece via o ambiente. Com isto, os robôs conseguem trabalhar paralelamente na solução de um problema sem precisar ter uma comunicação direta entre eles. O termo estigmergia foi introduzido pelo biólogo francês Pierre-Paul Grassé em 1959 (GRASSÉ, 1959), em um estudo sobre o comportamento de cupins na construção de seus ninhos. O princípio da estigmergia diz que o traço deixado no ambiente por qualquer ação estimula o desempenho de outra ação, pelo mesmo agente, ou por agentes diferentes. Considerando isto, em (BECKERS; HOLLAND; DENEUBOURG, 1994) foi desenvolvido um sistema multi-robô de coleta de objetos espalhados dentro de uma área bem definida.

Os robôs usados são equipados com uma pinça que permite segurar os discos, e dois sensores infravermelhos usados para evitar obstáculos. Além de segurar os discos, a pinça é usada como um sensor mecânico que indica a resistência ao movimento do disco: se o disco apresentar uma alta resistência a ser movimentada, isso determina uma alta densidade de discos, em contrapartida, se o disco apresentar uma baixa resistência, isso indica uma baixa densidade. Os robôs utilizados não possuem uma comunicação direta com os outros e não possuem de nenhum conhecimento sobre sua posição no ambiente. Cada robô é programado para identificar os obstáculos presentes no ambiente, sendo considerados como obstáculos os outros robôs e as paredes do ambiente. Na maior parte do tempo, o robô trabalha de maneira paralela, sem ser afetado pela interação direta com os demais robôs. Porém seu comportamento individual é influenciado pelo comportamento prévio dos outros robôs com os discos ou os grupos de discos já formados. Os robôs têm 3 comportamentos, e só um deles é ativado por vez:

1. O comportamento padrão do robô é movimentar-se em linha reta até detectar um obstáculo ou até ativar o micro-interruptor.

2. Quando é detectado um obstáculo, o robô realiza um comportamento de desvio de obstáculos. O robô gira em um ângulo aleatório até ficar afastado do obstáculo. Depois, é retomado o comportamento padrão. Se o robô estiver empurrando discos, enquanto realiza o comportamento de desvio de obstáculos, os discos são retidos na pinça, girando junto com o robô.

3. Quando a pinça empurra 3 ou mais discos o micro-interruptor faz retroceder o robô, liberando os discos. Depois, o robô gira em um ângulo aleatório e retoma o comportamento padrão.

Nas experiências foram utilizados 81 discos como elementos a serem agrupados, espalhados numa área de $250 \times 250 cm^2$. O número de robôs utilizados variam de 1 a 5, sendo 3 o número robôs ótimo para realizar o agrupamento dos discos. Quando o sistema é formado de um número superior a 3 robôs, estes ficavam mais tempo evitando uns aos outros, do que agrupando discos. Este estudo demostrou a eficácia do uso da estigmergia como uma forma de controle e coordenação para dividir as tarefas a serem realizadas por um grupo de robôs.

Continuando com esta abordagem, em (KAZADI; ABDUL-KHALIQ; GOODMAN, 2002) é analisada a influência da interação física dos robôs como agentes que executam o agrupamento e discos como elementos a serem agrupados. Os agentes são elementos ativos que apanham e movimentam discos. Os discos, chamados de *pucks*, são elementos passivos que podem ser apanhados e movimentados de um lugar a outro. Para realizar a análise, é simulado um sistema *não-físico*, onde não são consideradas as estruturas físicas que formam os robôs nem os *pucks*. Este sistema *não-físico* visa analisar somente o comportamento das regras básicas para agrupar, sem ter as restrições dos elementos físicos. Os resultados obtidos permitiram concluir que o aumento da quantidade de elementos já clusterizados acelera a clusterização dos elementos restantes.

Usando a mesma abordagem apresentada em (BECKERS; HOLLAND; DENEUBOURG, 1994), além de melhorar os sensores utilizados, em (VARDY; VOROBYEV; BANZHAF, 2014) é desenvolvido um método para agrupar elementos similares, mas de cores diferentes. O método acrescenta uma memória que auxilia o robô em localizar e regressar a locais já visitados. Cada robô possui uma câmera que consegue visualizar os discos em uma área próxima. Depois de ter capturado a imagem, o robô consegue identificar os *clusters* já formados, discos isolados e discos transportados por outro robô. Na memória de cada robô é armazenada a localização dos *clusters* com maior número de elementos. Para realizar os testes foram usados 4 robôs com a capacidade de transportar um disco por vez. As experiências foram realizadas com 40 discos espalhados em uma área de $187 \times 187\, cm^2$. Foram realizadas simulações para 1, 2, 4 e 8 tipos diferentes de discos e implementações em robôs reais para 1 e 2 tipos diferentes de discos. Este trabalho acrescentou o número

de grupos a serem formados além de melhorar o tempo de convergência apresentado no trabalho de (BECKERS; HOLLAND; DENEUBOURG, 1994).

O algoritmo, desenvolvido em (VARDY; VOROBYEV; BANZHAF, 2014), inicializa com um comportamento errante do robô, enquanto procura *clusters* pequenos. Quando o robô encontra um disco alvo, é calculada a distância entre o disco encontrado e o local a ser transportado. Se o disco está visualmente cerca a um *cluster*, este disco é descartado e é procurado outro disco. Caso contrário, o disco é coletado. Depois de coletar o disco, este é identificado pela sua cor e empurrado até o espaço predefinido para formar o *cluster* correspondente. Uma vez liberado o disco, o robô retrocede e volta para uma posição inicial. Durante todos os processos de movimentação, o robô pode realizar uma rotina para evitar obstáculos.

2.1.2 Clusterização de Robôs

Visando realizar um estudo de comportamentos bio-inspirados, em (GARNIER et al., 2005) implementam a conduta das baratas em enxames de 10 e 20 robôs. Neste trabalho é abordado um novo comportamento coletivo que é baseado em um agrupamento auto-organizado de robôs. Com regras comportamentais simples, o enxame de robôs consegue-se agrupar em áreas obscuras. Simulando as baratas, os robôs, além de procurar áreas menos iluminadas, procuram áreas com maior número de robôs. Esta regra emula a *tigmotaxia*, ou seja, a resposta comportamental inata de um organismo ao contato físico. Nas baratas a tigmotaxia acontece quando as antenas de uma barata tocam as antenas de outra barata. Nos robôs a tigmotaxia acontece por meio da comunicação infravermelha. Implementando regras básicas, este trabalho demostra que é possível obter comportamentos auto-organizáveis e de agrupamento através de decisões coletivas, utilizando uma comunicação semi-direta.

Baseados nas técnicas de clusterização de fichas em áreas espacialmente limitadas, em (LEE; KIM; KAZADI, 2005) é desenvolvida uma técnica de clusterização de robôs móveis. De maneira descentralizada, os robôs conseguem se agrupar e permanecer unidos, baseando-se unicamente em informação local e a estigmergia. Para este trabalho são simulados robôs com sensores infravermelhos e câmara de vídeo. Os robôs podem realizar duas ações iniciais: ficar imóvel, esperando a chegada de outro robô para começar um *cluster*, ou ficar em movimento até encontrar robôs imóveis. Depois de iniciar os primei-

ros *clusters*, os robôs podem se mover procurando um *cluster* com maior número de robôs. A avaliação do *cluster* encontrado é realizada por meio da visão do robô. Utilizando o campo visual do robô é possível estabelecer uma medida do tamanho do *cluster*. Em outros termos, a visão permite determinar a densidade do *cluster*. Dependendo da densidade do *cluster*, o robô permanece como membro do grupo ou se movimenta procurando um melhor grupo, sendo considerado melhor o *cluster* mais denso. A principal desvantagem deste tipo de clusterização é a impossibilidade de múltiplos *clusters*, devido à aderência entre os robôs.

Já em (GROSS; MAGNENAT; MONDADA, 2009), é utilizado o efeito das *Castanhas do Brasil*, descrito em (ROSATO et al., 1987), para realizar a segregação dos robôs. Usando esta técnica é possível separar dois ou mais tipos de elementos diferenciados pelos seus tamanhos. Ao separar os diferentes tipos de elementos consegue-se agrupá-los por tamanho. Baseando-se neste estudo, em (CHEN et al., 2012), é implementado este efeito usando robôs do tipo *e-Puck*. Estes representam os elementos a serem segregados. Como esta técnica requer elementos de diferentes tamanhos, os robôs são programados para que, dependendo de seu identificador, estabeleçam um *corpo virtual*. Ou seja, por meio da distância do alcance da comunicação é determinado um raio virtual que limita a aproximação de um robô de outro. Para implementar esta técnica nos robôs, foi preciso emular três componentes: a atração gravitacional, emulado por um ponto de atração comum para todos os robôs; a vibração, emulada pelo movimento aleatório dos robôs; e o efeito de colisão entre os elementos, emulado por uma repulsão entre os robôs próximos. A efetividade desta técnica é estabelecida pela diferença entre os tamanhos dos diferentes elementos. Quando a diferença entre os tamanhos é grande, a separação dos grupos é mais evidente.

Em (DI CARO; DUCATELLE; GAMBARDELLA, 2012), é desenvolvida uma técnica de clusterização espacial de robôs, onde os robôs só precisam de um conhecimento local. Por meio de troca de fichas virtuais, os robôs conseguem se agrupar. Esta estratégia proporciona um aumento de velocidade de clusterização e uma redução do gasto de energia, pois os robôs só precisam se comunicar para trocar informação, não precisando se movimentar. No trabalho de (DI CARO; DUCATELLE; GAMBARDELLA, 2012) é implementado o uso de só um tipo de ficha e, portanto, é apresentado o algoritmo só para dois *clusters*. Esta dissertação trata de estender esta técnica de clusterização espacial para mais de 2 classes.

2.2 Clusterização em Redes

As WSNs são redes de sensores sem fio desenvolvidas visando monitorar condições ambientais como temperatura e sons, entre outras (ABBASI; YOUNIS, 2007). Os sensores utilizados têm a capacidade de processar e transmitir a informação coletada. Estas redes de sensores têm grande aplicação em locais de difícil acesso ou áreas perigosas. Devido à grande dificuldade na substituição da bateria dos sensores, o consumo de energia é um fator crítico, precisando de protocolos eficientes para prolongar a vida útil de todos os nós da rede. Outro ponto importante em WSN é a tolerância a falhas, sendo necessários algoritmos de roteamento e técnicas para auto-organização dos dispositivos dinâmicos. Além disso, nas aplicações de WSN é utilizado um grande número de sensores (centenas e milhares). Para conseguir lidar com este número de sensores, é requerida uma arquitetura escalável e uma estratégia eficiente de coordenação. Com o propósito de atingir o objetivo de escalabilidade são desenvolvidas técnicas de clusterização de sensores. Estas técnicas iniciam procurando uma estação base local, chamada de *Cluster-Head*, e a partir desta são determinados os membros do grupo. A estação base tem a função de otimizar a agregação e roteamento de dados. Algumas destas técnicas serão explicadas brevemente em seguida.

Em (HEINZELMAN; CHANDRAKASAN; BALAKRISHNAN, 2000), é proposto um protocolo de clusterização hierárquica adaptativa de baixa energia, chamada de LEACH (*Low-Energy Adaptative Clustering Hierarchy*). Este protocolo utiliza uma rotação aleatória da localização da estação base para distribuir eventualmente a carga de energia entre os sensores da rede. Quando a estação base é fixa e selecionada a priori, a vida útil da estação base se esgota muito rápido. O principal objetivo do LEACH é rotear a estação base para que não seja esgotada a bateria de um único sensor. A eleição da estação base local é realizada de forma autônoma, a qualquer momento, obedecendo a certa probabilidade. Cada sensor estabelece a qual estação base pertencer, determinada pelo menor consumo de energia durante a comunicação.

As redes *ad hoc* móveis, MANETs, fornecem um método de estabelecer uma comunicação entre os nós dispersos da rede, sem a necessidade de uma infraestrutura física (SAKHAEE; JAMALIPOUR, 2008). Em redes de grande escala, as MANETs são ineficientes. Isto é devido ao maior número de interconexões e as despesas gerais de processamento. A clusterização deste tipo de redes pode proporcionar um sistema com baixos custos relacionados à manutenção, eficiência energética e distribuição de carga de trabalho em uma

rede.

Em (BASU; KHAN; LITTLE, 2001), é proposto um algoritmo distribuído de clusterização, chamado de MOBIC *(Mobility Based Metric for Clustering)*, baseado em uma métrica de mobilidade para redes *ad hoc* móveis. A métrica quantifica o movimento de um nó baseando-se na relação de velocidade entre os nós vizinhos. Para isso, é medida a distância entre um nó e outro nó em dois instantes de tempo diferentes. A distância é medida detectando o nível da potência dos sinais. Depois de determinar a medida de mobilidade, cada nó compartilha sua informação com a vizinhança. Posteriormente, é selecionado como *cluster-head* quem tiver a menor mobilidade, e todos os nós vizinhos considerando dois saltos são identificados como membros do *cluster*.

2.3 Considerações Finais do Capítulo

Neste capítulo foram apresentados resumidamente diversos trabalhos sobre clusterização em exame de robôs, sendo estes utilizados como agentes ativos para realizar a clusterização de elementos passivos, e também como elementos auto-organizáveis, gerando grupos de robôs estáveis. Também foram apresentados alguns trabalhos relacionados às redes de sensores sem-fio e redes de *ad hoc* móveis. O capítulo seguinte apresenta detalhadamente o algoritmo de clusterização espacial para exames de robôs proposto, implementado e avaliado nesta dissertação.

Capítulo 3

ALGORITMO DE AGRUPAMENTO ESPACIAL

ESTE capítulo apresenta o algoritmo de clusterização espacial de robôs generalizado para duas ou mais classes. O algoritmo proposto é baseado na ideia do algoritmo distribuído para duas classes apresentado em (DI CARO; DUCATELLE; GAMBARDELLA, 2012). Este algoritmo não requer movimentação dos robôs para identificar e formar os *clusters*. Os robôs utilizam o conhecimento adquirido via troca de mensagens com os robôs vizinhos. Um robô é dito *vizinho* de outro robô r, quando se encontra dentro da circunferência ao redor do robô r, de um raio limitado pelo alcance da comunicação entre eles.

O algoritmo de clusterização baseia-se principalmente no movimento de fichas virtuais, chamadas de *cargas*, pelos robôs. Estas cargas podem ser *dinâmicas* ou *estáticas*. A carga estática determina a classe do robô e as cargas dinâmicas representam o movimento de uma carga no enxame de robôs. Cada um dos robôs do enxame pode ter uma única carga estática, mas qualquer número de cargas dinâmicas. Se um robô possuir uma carga estática, este é denominado como *carregado*. Por outro lado, se um robô não possuir nenhuma carga estática, este é denominado *descarregado*.

Para todo robô i do enxame, uma variável de estado local u_i indica a classe da carga estática deste robô. Note que a classe identifica o *cluster*. Na técnica usada em (DI CARO; DUCATELLE; GAMBARDELLA, 2012) esta variável é binaria, pois só é preciso determinar se o robô está *carregado* ou *descarregado*. Para generalizar o algoritmo para duas ou mais classes a variável u_i é redefinida como $u_i \in \{0, 1, 2, ..., \zeta - 1\}$, sendo ζ o número de classes. Note que o robô i está descarregado quando $u_i = 0$ e carregado quando $u_i \neq 0$. Com esta generalização, esta variável é também usada como um indicador do peso da classe. Como

antes, cada robô do enxame pode ter uma única carga estática, mas lidar com qualquer número de cargas dinâmicas.

Na Figura 6 é apresentado um diagrama ilustrativo do robô i, incluindo a carga estática e diferentes cargas dinâmicas. No diagrama, ℓ representa o identificador da carga dinâmica, sendo $\ell \in \{1, 2, \ldots, d_i\}$, d_i o número de cargas dinâmicas presentes no robô i, e $c_{i\ell}$ representa a classe da carga dinâmica ℓ, com $c_{i\ell} \in \{1, 2, \ldots, \zeta - 1\}$, $f_{i\ell}$ a fase correspondente da carga ℓ, com $f_{i\ell} \in \{Fase_2, Fase_3, Fase_4, Fase_5\}$. Na Seção 3.1 serão detalhadas as fases das cargas.

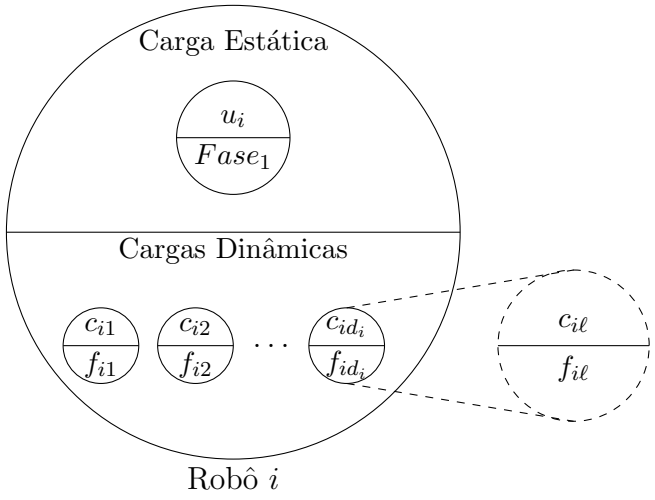


Figura 6: Diagrama ilustrativo de um robô, a carga estática e as cargas dinâmicas

Para poder guiar os movimentos das cargas, é usada outra variável de estado $v_i \in [0, 1]$ local a cada robô i. Esta variável permite determinar a densidade local das cargas estáticas, usando um *filtro de consenso distribuído* (OLFATI-SABER; SHAMMA, 2005), de tal forma que, é possível estabelecer se a carga estática presente no robô i continuará estática ou será movimentada. Em (DI CARO; DUCATELLE; GAMBARDELLA, 2012), a atualização da densidade é realizada conforme a Equação 2:

$$v_i^+ = v_i + \delta \left(\sum_{j \in \mathcal{N}_i} (v_j - v_i) + \sum_{j \in J_i} (u_j - v_i) \right), \tag{2}$$

onde v_i^+ é o valor atualizado de v_i, $\mathcal{N}_i$ é o conjunto de vizinhos do robô i, $J_i = \mathcal{N}_i \bigcup i$, e δ é uma constante, tal que $0 \leq \delta \leq \frac{1}{n}$, n sendo o número total de robôs do enxame. A Equação 2 permite aproximar a densidade do robô i da densidade dos robôs na sua vizinhança. Note que, a variável u_i para uma clusterização em 2 classes está restrita a 0 ou 1, permitindo manter a densidade de carga estática no intervalo $[0, 1]$.

Para estender o algoritmo de agrupamento espacial para mais de 2 classes, a variável u_i não estará mais restrita aos valores $\{0, 1\}$. Portanto, é preciso modificar a Equação 2, normalizando o valor de u_i e assim manter as densidades calculadas dentro da faixa $[0, 1]$. Assim a atualização da densidade no algoritmo proposto é realizada conforme a Equação 3:

$$v_i^+ = v_i + \delta \left(\sum_{j \in \mathcal{N}_i} (v_j - v_i) + \sum_{j \in J_i} \left(\frac{u_j}{\zeta - 1} - v_i \right) \right). \tag{3}$$

O seguinte exemplo permite uma melhor compreensão da atualização de densidade. Para isso, suponha que o robô de identificador 1 tenha uma vizinhança $\mathcal{N}_i \in \{2, 3, 4, 5\}$, conforme mostra a Figura 7, onde os robôs são representados pelos círculos grandes e as cargas pelos círculos pequenos. Além disso, assuma que os robôs 1 e 2 estão carregados com cargas estáticas de peso 2 e possuem uma densidade de 0,551 e 0,723 respectivamente, os robôs 3 e 4 estão carregados com cargas estáticas de peso 1 e possuem uma densidade de 0,503 e 0,612 respectivamente e o robô 5 está descarregado com uma densidade de 0,365. Se o tamanho do enxame é de 20 robôs e o número de classes é 3, então o próximo valor de densidade do robô 1 será de 0,563.

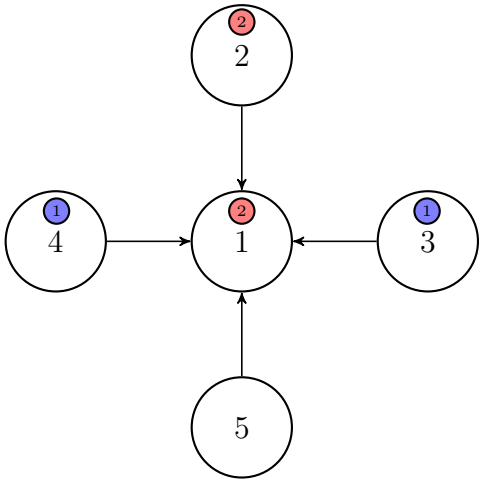


Figura 7: Exemplo ilustrativo da vizinhança e atualização da densidade de cargas estáticas

Para chegar ao resultado, considere a Tabela 1 que resume alguns dos cálculos necessários para obter o próximo valor da densidade. Para isso, sejam T_1 e T_2 os resultados dos somatórios:

$$T_1 = \sum_{j \in \mathcal{N}_i} (v_j - v_i) = -0,001 \tag{4}$$

$$T_2 = \sum_{j \in J_i} \left(\frac{u_j}{\zeta - 1} - v_i \right) = 0,245. \tag{5}$$

Tabela 1: Cálculos e resumo de dados do exemplo

j	2	3	4	5
u_j	2	1	1	0
v_j	0,723	0,503	0,612	0,365
$v_j - v_i$	0,172	-0,048	0,061	-0,186
$\frac{u_j}{\zeta-1} - v_i$	0,449	-0,051	-0,051	-0,551

Finalmente, o próximo valor de densidade do robô 1 é conforme Equação 6:

$$v_1^+ = 0,551 + 0,05(T_1 + T_2) = 0,563, \tag{6}$$

onde, $\delta = 0,05$.

A variável local v_i, além de guiar os movimentos das cargas dentro do enxame, também cria faixas de densidade relacionadas ao peso da carga. Ao considerar unicamente duas classes, em (DI CARO; DUCATELLE; GAMBARDELLA, 2012) estabelece-se que o valor para separar as faixas é de v_i = 0,5. Quando o número de classes aumenta, é preciso estabelecer faixas de densidades de acordo com o número de classes requeridas durante o processo de clusterização. Portanto, são calculados os valores máximo e mínimo da densidade para cada classe, conforme a Equação 7:

$$\frac{u_i}{\zeta} < v_i \leq \frac{u_i + 1}{\zeta}. \tag{7}$$

Note que as ζ classes são definidas pelas faixas de densidade $]0, \frac{1}{\zeta}],]\frac{1}{\zeta}, \frac{2}{\zeta}], \ldots,]1 - \frac{1}{\zeta}, 1]$. Na Figura 8, é apresentado um exemplo ilustrativo do resultado esperado do algoritmo de clusterização para ζ classes, onde cada círculo representa um robô do enxame e as cores representam os grupos de robôs da mesma classe.

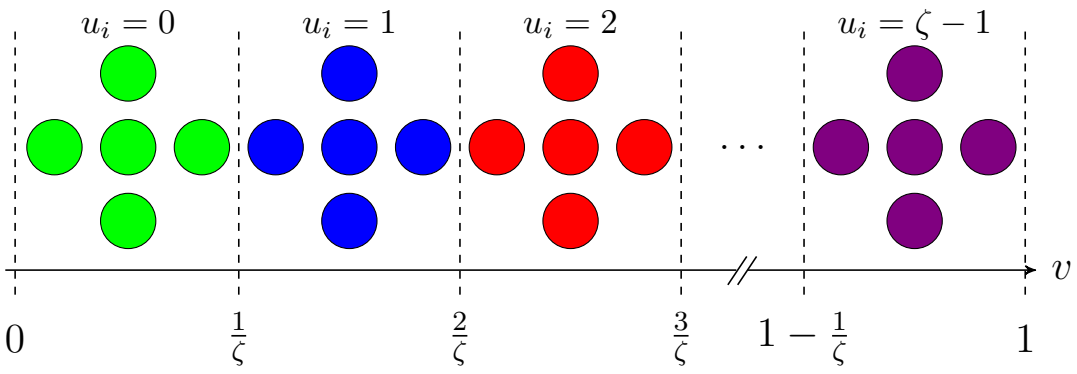


Figura 8: Exemplo ilustrativo de clusterização em ζ classes

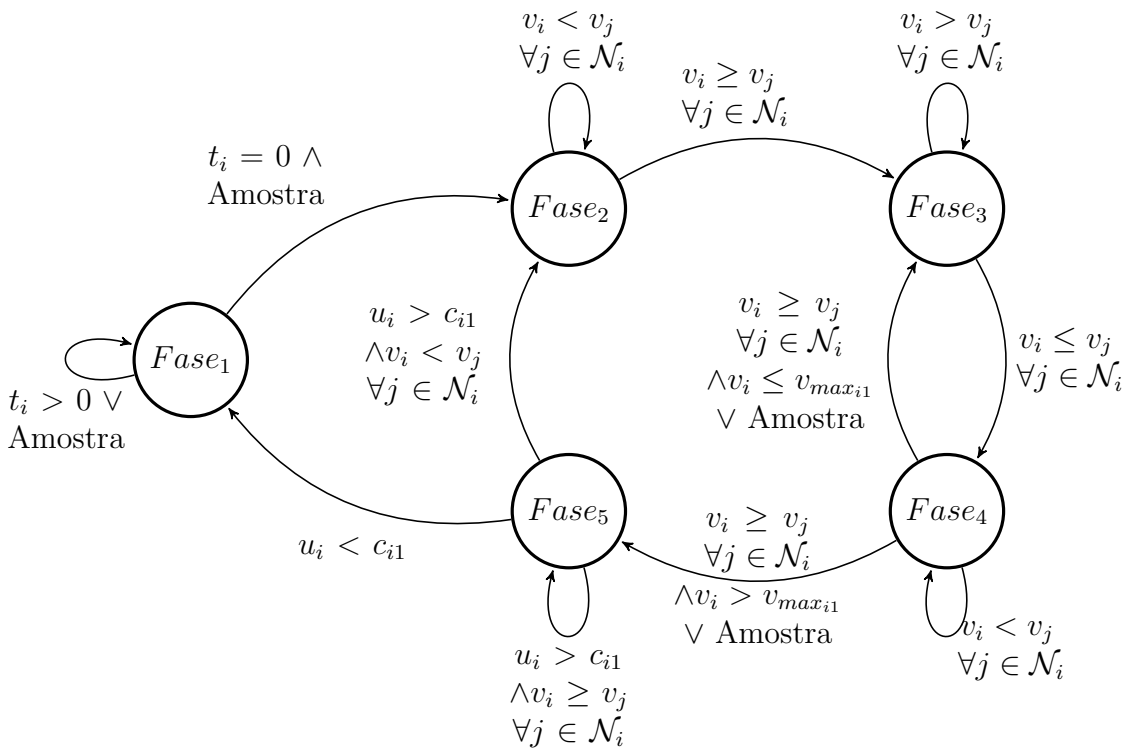


Figura 9: Diagrama de transição de fase da carga

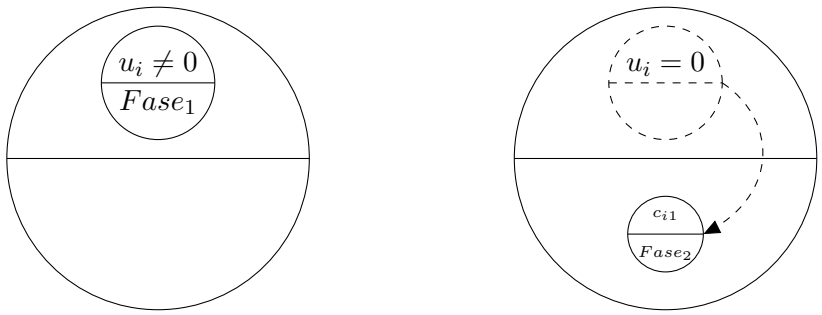
3.1 Fases de Clusterização

Durante o processo de clusterização, o comportamento de cada uma das cargas tanto estática quanto dinâmicas é regido por cinco fases distintas que permitem que a carga percorra o enxame em procura da região de maior densidade. É evidente que os robôs descarregados e que não possuam cargas dinâmicas não realizam nenhuma destas fases, este caso será explicado na Seção 3.2. Vale ressaltar que, no começo de cada fase, é realizada a recepção de novas informações provenientes dos vizinhos. Para isso, são usadas as primitivas *receber* e *enviar* cujos funcionamentos serão explicados no Capítulo 4, junto com outros detalhes de implementação. A Figura 9 resume o diagrama de transição de fase da carga, detalhadas nas seções seguintes.

3.1.1 Início estacionário

Os detalhes desta fase estão descritos no Algoritmo 1. Esta fase é gerenciada somente quando a carga é estática, indicada por $u_i \neq 0$, ou seja, quando o robô está carregado. Nesta fase, a carga é retida no robô, e só deixa de ser estática se certo período de tempo

T pré-definido esgota-se. Para isso, um contador t_i é inicializado de acordo com o tempo de espera T, e decrementado somente nos casos em que a densidade fique fora da faixa relacionada com a classe conforme definido na Equação 7. Isto permite um tempo de espera para que o robô carregado compartilhe seu conhecimento sobre o ambiente podendo aumentar o valor da densidade da sua região. Note que, quando o valor da densidade está dentro da faixa certa, o contador t_i não será decrementado e, portanto, o robô não movimentará a carga estática presente nele. Quando t_i chega a 0, *i.e.* termina o período de espera, a carga é movimentada com uma certa probabilidade ou amostra. Esta probabilidade pode ocasionar um maior tempo de espera para que o robô compartilhe seu conhecimento com os robôs vizinhos. Depois disso, a carga deixa de ser estática e passa a ser dinâmica, incrementando o contador d_i. Este contador registra o número de cargas dinâmicas que o robô i possui até o presente momento. Em seguida, uma lista L_i formada pelas variáveis $c_{i\ell}$ e $f_{i\ell}$, $1 \geq \ell \geq d_i$, registra os valores correspondentes às classes e fases relacionadas com as d_i cargas dinâmicas. A lista L_i é gerenciada por cada robô i. Note que a variável c registra o valor de u_i, e a variável f registra o valor da próxima fase, neste caso $Fase_2$. Depois, as variáveis c e f são inseridas na lista L_i. Por último, é atribuído o valor 0 à variável u_i. Na Figura 10 é representada a substituição da carga estática por uma carga dinâmica. Nas seguintes seções, a carga dinâmica tratada em cada fase correspondente é a que se recupera da primeira posição da lista L_i de cargas dinâmicas, ou seja, c_{i1}.



(a) Robô i carregado com $d_i = 0$ (b) Robô i descarregado com $d_i = 1$

Figura 10: Ilustração dos eventos que acontecem na $Fase_1$

Algoritmo 1 $Fase_1$: Inicio estacionário no robô i

1: $t_i \leftarrow T$;
2: **enquanto** $u_i \neq 0$ **faça**
3: **para todo** $j \in \mathcal{N}_i$ **faça**
4: **receber** $\langle u_j, v_j, c_j, f_j \rangle$;
5: **se** $c_j \neq 0$ **então**
6: $d_i \leftarrow d_i + 1$;
7: **inserir** (c_j, f_j) em L_i;
8: **fim se**;
9: **fim para**;
10: **atualizar** v_i;
11: **se** $t_i > 0$ **então**
12: **se** $v_i < \frac{u_i}{\zeta} \wedge v_i \geq \frac{u_i+1}{\zeta}$ **então**
13: $t_i \leftarrow t_i - 1$;
14: **fim se**;
15: **senão se** critério de probabilidade atingido **então**
16: $d_i \leftarrow d_i + 1$;
17: $c \leftarrow u_i$;
18: $f \leftarrow Fase_2$;
19: **inserir** (c, f) em L_i;
20: $u_i \leftarrow 0$;
21: **fim se**;
22: **enviar** $\langle u_i, v_i \rangle$ para todos os robôs em $\mathcal{N}_i$;
23: **fim enquanto**;

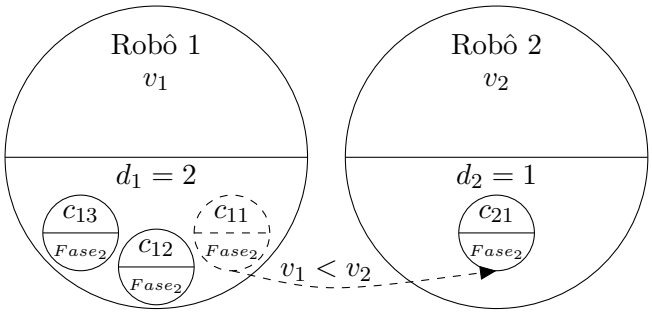
3.1.2 Primeiro movimento ascendente

Durante esta fase, detalhada no Algoritmo 2, uma carga dinâmica é movimentada de robô em robô, procurando sempre a região de maior densidade. Isto significa que o robô i envia a carga dinâmica para o vizinho $j \in \mathcal{N}_i$ com maior valor de densidade. No Algoritmo 2, a função $g(\mathcal{N}_i)$ permite identificar o robô k com maior densidade na vizinhança do robô i. O objetivo desta movimentação é chegar até o centro do *cluster* formado pelos robôs já carregados. Quando a carga ascende até um máximo local, ou seja, até o robô i com densidade $v_i \geq v_k$, o valor da variável v_i é registrado na variável $v_{max_{i1}}$. Esta variável de densidade máxima encontrada pela carga faz parte da informação associada à carga dinâmica e é transmitida junto com a informação de classe $c_{i\ell}$ e fase $f_{i\ell}$, sendo ℓ o identificador da carga dinâmica. Este valor será usado na $Fase_4$. Depois disso, a carga muda para $Fase_3$ e é movimentada para um vizinho aleatório.

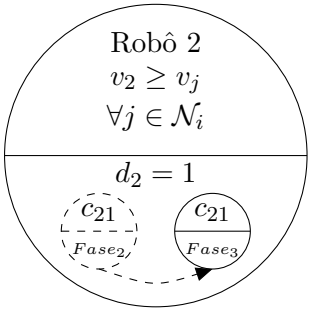
Na Figura 11 é apresentado um exemplo ilustrativo dos eventos que acontecem na $Fase_2$. Ao trabalhar com cargas de diferentes pesos e diferentes fases, foram dadas prioridades na ordem de armazenamento na lista de cargas dinâmicas L_i, primeiro pela

Algoritmo 2 $Fase_2$: Movimento ascendente no robô i

1: **recuperar** (L_i);
2: **para todo** $j \in \mathcal{N}_i$ **faça**
3: **receber** $\langle u_j, v_j, c_j, f_j \rangle$;
4: **se** $c_j \neq 0$ **então**
5: $d_i \leftarrow d_i + 1$;
6: **inserir** (c_j, f_j) em L_i;
7: **fim se**;
8: **fim para**;
9: **atualizar** v_i;
10: $k \leftarrow g(\mathcal{N}_i)$;
11: **se** $v_i < v_k$ **então**
12: **enviar** $\langle c_{i1}, f_{i1} \rangle$ para o robô k;
13: **senão**
14: $v_{max_{i1}} \leftarrow v_i$;
15: $f_{i1} \leftarrow Fase_3$;
16: $r \leftarrow aleatorio(\mathcal{N}_i)$;
17: **enviar** $\langle c_{i1}, f_{i1}, v_{max_{i1}} \rangle$ para o robô r;
18: **fim se**;
19: $d_i \leftarrow d_i - 1$;
20: **remover** (L_i);
21: **enviar** $\langle u_i, v_i \rangle$ para todos os robôs em $\mathcal{N}_i$;



(a) Exemplo da transferência da uma carga entre dois robôs



(b) Exemplo da transição da $Fase_2$ para $Fase_3$

Figura 11: Ilustração dos eventos que acontecem na $Fase_2$

maior fase $f_{i\ell}$ e depois pelo maior peso $c_{i\ell}$. Estas prioridades facilitam o processo de clusterização quando existe uma grande quantidade de cargas dinâmicas.

3.1.3 Movimento descendente

Os detalhes deste processo estão descritos no Algoritmo 3. Ao contrário da fase de movimento ascendente, durante esta fase uma carga dinâmica é movimentada de robô em robô, procurando sempre a região com a menor densidade entre os vizinhos. No Algoritmo 3 a função $h(\mathcal{N}_i)$ permite identificar o robô k com menor densidade presente na vizinhança do robô i. O motivo deste movimento é reiniciar a busca pela região de maior densidade, permitindo que a carga não fique presa em uma região cuja densidade é um máximo local. Quando a carga desce até um mínimo local, e assumindo que o robô i representa este mínimo local, ou seja, $v_i \leq v_k$, a carga transita para a $Fase_4$ e é movimentada para um vizinho aleatório.

Algoritmo 3 $Fase_3$: Movimento descendente no robô i

1: **recuperar** (L_i);
2: **para todo** $j \in \mathcal{N}_i$ **faça**
3: **receber** $\langle u_j, v_j, c_j, f_j \rangle$;
4: **se** $c_j \neq 0$ **então**
5: $d_i \leftarrow d_i + 1$;
6: **inserir** (c_j, f_j) em L_i;
7: **fim se**;
8: **fim para**;
9: **atualizar** v_i;
10: $k \leftarrow h(\mathcal{N}_i)$;
11: **se** $v_i < v_k$ **então**
12: **enviar** $\langle c_{i1}, f_{i1}, v_{max_{i1}} \rangle$ para o robô k;
13: **senão**
14: $f_{i1} \leftarrow Fase_4$;
15: $r \leftarrow aleatorio(\mathcal{N}_i)$;
16: **enviar** $\langle c_{i1}, f_{i1}, v_{max_{i1}} \rangle$ para o robô r;
17: **fim se**;
18: $d_i \leftarrow d_i - 1$;
19: **remover** (L_i);
20: **enviar** $\langle u_i, v_i \rangle$ para todos os robôs em $\mathcal{N}_i$;

3.1.4 Segundo movimento ascendente

Os detalhes deste processo estão descritos no Algoritmo 4. Semelhante ao comportamento da $Fase_2$, nesta fase uma carga dinâmica é movimentada de robô em robô até encontrar

uma região com um valor de densidade alto. Quando a carga chega a um máximo local de densidade, ou seja, quando $v_i \geq v_k$, este valor é comparado com o valor máximo de densidade $v_{max_{i1}}$ alcançado na $Fase_2$. Se o novo máximo local é maior que o valor armazenado, então a carga muda para $Fase_5$, continuando o processo de clusterização. Senão, *i.e.* se o valor de v_i não é maior, a carga mudará igualmente para a $Fase_5$ com certa probabilidade p ou para $Fase_3$ com $1 - p$, com a intenção de encontrar uma nova região carregada com maior densidade. A probabilidade p é determinada de forma empírica, pois esta probabilidade permite que o processo de clusterização continue na procura de uma região com maior densidade ou continue a clusterização em uma região de menor de densidade em comparação com a encontrada na $Fase_2$. Depois de mudar de fase, a carga é movimentada para um vizinho aleatório.

Algoritmo 4 Fase$_4$: Movimento ascendente no robô i

1: **recuperar** (L_i);
2: **para todo** $j \in \mathcal{N}_i$ **faça**
3: **receber** $\langle u_j, v_j, c_j, f_j \rangle$;
4: **se** $c_j \neq 0$ **então**
5: $d_i \leftarrow d_i + 1$;
6: **inserir** (c_j, f_j) em L_i;
7: **fim se**;
8: **fim para**;
9: **atualizar** v_i;
10: $k \leftarrow g(\mathcal{N}_i)$;
11: **se** $v_i < v_k$ **então**
12: **enviar** $\langle c_{i1}, f_{i1}, v_{max_{i1}} \rangle$ para o robô$_k$;
13: **senão**
14: **se** $v_i > v_{max_{i1}}$ **então**
15: $f_{i1} \leftarrow Fase_5$;
16: **senão se** critério de continuar ou procurar nova região **então**
17: $f_{i1} \leftarrow Fase_5$;
18: **senão**
19: $f_{i1} \leftarrow Fase_3$;
20: **fim se**;
21: $r \leftarrow aleatorio(\mathcal{N}_i)$;
22: **enviar** $\langle c_{i1}, f_{i1}, v_{max_{i1}} \rangle$ para o robô$_r$;
23: **fim se**;
24: $d_i \leftarrow d_i - 1$;
25: **remover** (L_i);
26: **enviar** $\langle u_i, v_i \rangle$ para $\mathcal{N}_i$;

Algoritmo 5 Fase$_5$: Movimento descendente lento no robô i

1: **recuperar** (L_i);
2: $v_k \leftarrow 0$;
3: **para todo** $j \in \mathcal{N}_i$ **faça**
4: **receber** $\langle u_j, v_j, c_j, f_j \rangle$;
5: **se** $c_j \neq 0$ **então**
6: $d_i \leftarrow d_i + 1$;
7: **inserir** (c_j, f_j) em L_i;
8: **fim se**;
9: **fim para**;
10: **atualizar** v_i;
11: **se** $u_i \geq c_{i1}$ **então**
12: $k \leftarrow g(\mathcal{N}_i) < v_i$;
13: **se** $v_k = 0$ **então**
14: $f_{i1} \leftarrow Fase_2$;
15: $r \leftarrow aleatorio(\mathcal{N}_i)$;
16: **envia** $\langle c_{i1}, f_{i1} \rangle$ para o robô$_r$;
17: **senão**
18: **envia** $\langle c_{i1}, f_{i1} \rangle$ para o robô$_k$;
19: **fim se**;
20: **senão se** $u_i = 0$ **então**
21: $u_i \leftarrow c_{i1}$;
22: **senão**
23: $temp \leftarrow u_i$;
24: $u_i \leftarrow c_{i1}$;
25: $c_{i1} \leftarrow temp$;
26: **fim se**;
27: $d_i \leftarrow d_i - 1$;
28: **remover** (L_i);
29: **enviar** $\langle u_i, v_i \rangle$ para todos os robôs em $\mathcal{N}_i$;

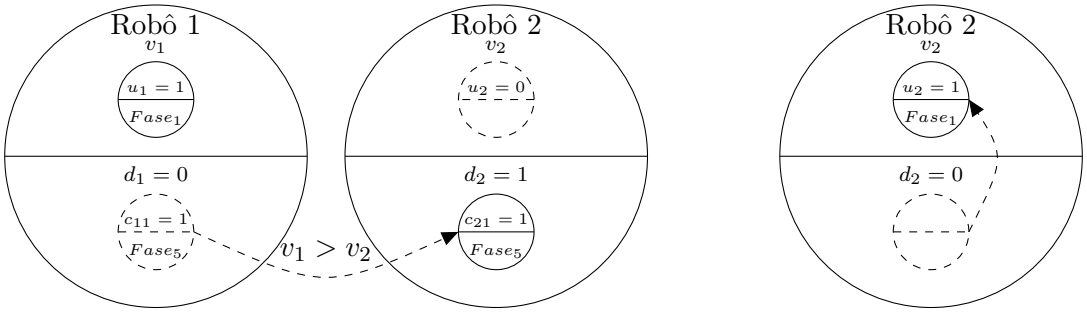
3.1.5 Movimento descendente lento

Os detalhes deste processo estão descritos no Algoritmo 5. O propósito geral desta fase é encontrar um robô para agrupá-lo, ou seja, encontrar o robô que deve mudar a carga dinâmica considerada para estática. Para isso, a carga dinâmica é movimentada de robô em robô, procurando o vizinho $j \in \mathcal{N}_i$ com a maior densidade v_j, tal que $v_j < v_i$, até chegar em um robô com $u_i = 0$ ou um robô com $u_i \neq 0$, tal que $u_i < c_{i1}$, onde c_{i1} identifica a classe da carga dinâmica que está sendo tratada. No caso em que a carga dinâmica encontre um robô descarregado, esta muda para $Fase_1$ e o robô fica então carregado com $u_i = c_{i1}$. No caso em que a carga dinâmica encontre um robô carregado, mas com $u_i < c_{i1}$, serão trocadas as cargas entre si. Em outros termos, a carga dinâmica passa a ser estática, mudando para $Fase_1$, enquanto a carga estática muda para dinâmica,

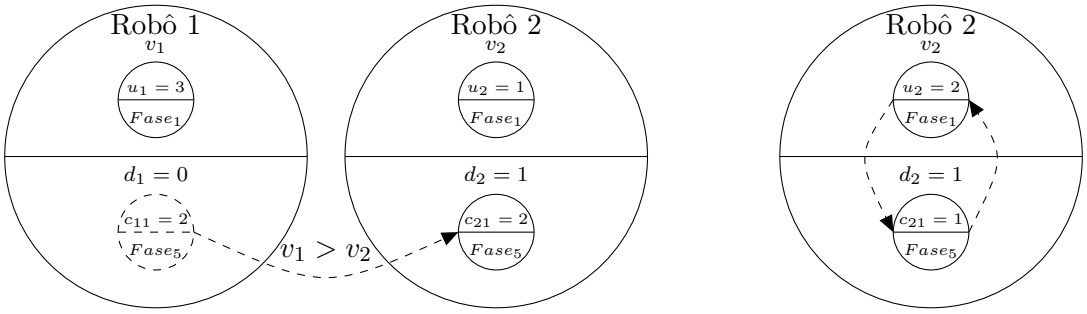
deixando o robô i carregado com a carga de maior peso. Isto proporciona uma prioridade maior na clusterização de cargas da maior classe. A carga de menor peso fica na $Fase_5$, continuando com o processo de clusterização. Se a carga chegar até um mínimo local e não conseguir encontrar nenhum robô descarregado ou um robô carregado com $u_i < c_{i1}$, a carga transita para $Fase_2$ e é movimentada para um vizinho escolhido aleatoriamente, recomeçando todo o processo de movimentação da carga dinâmica. Note que quando a carga muda para a $Fase_1$, o contador t_i que estabelece o tempo de espera é reiniciado. Na Figura 12 são apresentados dois exemplos do comportamento desta fase.

Note que, quando uma carga dinâmica é enviada pelo robô i, o número de cargas dinâmicas d_i é decrementado, e a carga dinâmica enviada é removida da lista de cargas dinâmicas L_i. Quando uma nova carga dinâmica é recebida pelo robô i, o contador d_i é incrementado, e a informação da carga é armazenada na lista de cargas dinâmicas. Este procedimento é repetido sempre que uma carga dinâmica é movimentada para outro robô.

Note que o comportamento descrito na $Fase_1$ acontece unicamente nos robôs carregados. Ademais os robôs com cargas dinâmicas, independentemente do valor da variável u_i, podem realizar as $Fase_2$ a $Fase_5$.



(a) A carga dinâmica chega a um robô descarregado



(b) A carga dinâmica chega a um robô carregado com $u_i < c_{i1}$

Figura 12: Exemplos das possíveis situações na $Fase_5$

3.2 Robôs descarregados sem cargas dinâmicas

Depois de apresentar as fases pelas quais a carga precisa passar para conseguir a clusterização, é necessário explicar o comportamento do algoritmo quando o robô i está descarregado e não possui nenhuma carga dinâmica. Conforme foi explicado anteriormente, o robô i é considerado descarregado quando $u_i = 0$. Este fato não significa que o robô deva ser visto como não clusterizado, pois um robô i com $u_i = 0$ pertence à classe 0. No entanto, nunca existirá uma carga dinâmica ℓ de peso $c_{i\ell} = 0$. O Algoritmo 6 detalha o comportamento do robô neste caso. Neste caso, embora o robô i não tenha cargas para enviar a seus vizinhos, este compartilha suas informações de densidade e classe, e também atualiza sua densidade à medida que vá recebendo informação deles. Quando o robô descarregado recebe uma carga dinâmica, esta será tratada de acordo com as fases descritas anteriormente.

Algoritmo 6 Robô i descarregado sem cargas dinâmicas

```
1: enquanto u_i = 0 ∧ d_i = 0 faça
2:     para todo j ∈ N_i faça
3:         receber ⟨u_j, v_j, c_j, f_j⟩;
4:         se c_j ≠ 0 então
5:             d_i ← d_i + 1;
6:             inserir (c_j, f_j) em L_i;
7:         fim se;
8:     fim para;
9:     atualizar v_i;
10:    enviar ⟨u_i, v_i⟩ para todos os robôs em N_i;
11: fim enquanto;
```

3.3 Considerações Finais do Capítulo

Neste capítulo foi apresentado o algoritmo de clusterização espacial de robôs para ζ classes. O algoritmo é distribuído, e permite a clusterização de robôs sem precisar movimentá-los fisicamente. Os robôs do enxame não precisam de um conhecimento global, necessitando somente do conhecimento adquirido dos vizinhos. Cargas estáticas espalhadas aleatoriamente nos robôs determinam uma densidade que guia a movimentação das cargas dinâmicas ao longo do enxame, que culmina com o agrupamento desejado. No próximo capítulo serão descritos os detalhes da implementação e analisados os resultados obtidos.

Capítulo 4

ASPECTOS DE IMPLEMENTAÇÃO E RESULTADOS

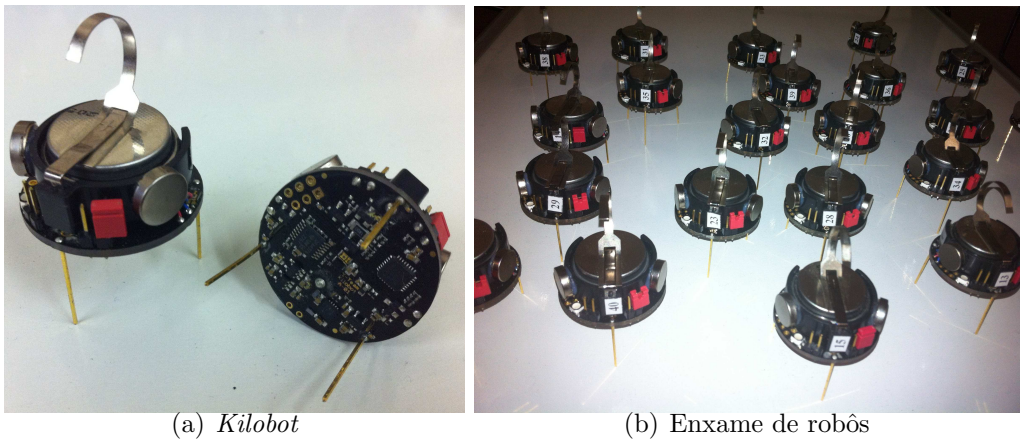
ESTE capítulo apresenta a implementação do algoritmo de clusterização espacial descrito no Capítulo 3 utilizando robôs reais. Também são analisados os resultados obtidos. A análise dos resultados permite avaliar a eficácia do algoritmo em realizar um agrupamento balanceado de robôs.

Na Seção 4.1 são apresentados os detalhes do robô e dos enxames utilizados na implementação, juntamente com os detalhes da comunicação realizada entre os robôs do enxame. A Seção 4.2 detalha a composição da mensagem utilizada durante o processo de comunicação necessária. A Seção 4.3 apresenta algumas considerações na implementação do algoritmo no robô. A Seção 4.4 aborda os conceitos de *separabilidade linear* e *desequilíbrio de classe* usados para analisar o desempenho da implementação proposta. A Seção 4.5 apresenta os resultados obtidos. A Seção 4.6 apresenta as considerações finais para o capítulo.

4.1 O Robô

Para a implementação do algoritmo proposto foram utilizados enxames de robôs do tipo *Kilobot* (RUBENSTEIN; AHLER; NAGPAL, 2012), mostrados na Figura 13. O *Kilobot* é um robô de $33\,mm$ de diâmetro e $34\,mm$ de altura. O funcionamento dos dispositivos embarcados é gerenciado por um microprocessador Atmel ATmega328p de $8\,MHz$ (ATMEL, 2013). O robô tem uma memória *Flash* de $32\,KB$ para o armazenamento do programa a ser executado, uma SRAM (*Static Random Access Memory*) de $2\,KB$ e uma memória EE-

PROM (*Electrically-Erasable Programmable Read-Only Memory*) de 1 KB que armazena dados de calibração e dados não-voláteis como o identificador único para cada robô.



(a) *Kilobot* (b) Enxame de robôs

Figura 13: Robôs do tipo *Kilobot*

O *Kilobot* está equipado com um LED RGB capaz de produzir qualquer cor através da combinação de intensidade das cores vermelho, verde e azul. A locomoção do robô é permitida por dois motores de vibração em forma de moeda, que utilizam a força centrípeta produzida pela vibração para gerar o movimento.

Cada robô do enxame é designado por um único identificador, armazenado na memória EEPROM. Este identificador é necessário durante a comunicação do robô com os demais robôs da vizinhança. Para realizar a comunicação, todos os robôs contam com um transmissor e receptor infravermelhos. Estes são localizados na parte central na parte inferior do robô. O transmissor tem uma emissão isotrópica e o receptor tem uma recepção padrão, proporcionando aos robôs receber mensagens homogeneamente de todas as direções dentro de um raio de 10 cm. Quando o transmissor está ativado, qualquer robô na vizinhança pode receber o sinal refletido na superfície. Isto determina um raio máximo de comunicação de 10 cm.

4.2 Comunicação

Para a implementação do algoritmo de clusterização, cada robô do enxame realiza dois tipos de comunicações: uma comunicação *broadcast local*, de tal forma que a informação é compartilhada com toda a vizinhança; e uma comunicação *ponto a ponto*, por meio da qual é transmitida a carga de um robô a outro da vizinhança. Por meio da comunicação infravermelha, o *Kilobot* consegue transmitir uma mensagem composta de 23 *bits* no má-

ximo. Na Figura 14 é mostrada a estrutura da mensagem enviada. Os primeiros 3 *bits* definem o tipo de mensagem enviada. Os 5 *bits* seguintes contêm o identificador do robô emissor da mensagem. Dependendo do tipo de mensagem, os 8 *bits* seguintes conterão a informação da densidade do robô emissor ou a densidade máxima encontrada por uma carga. Os 2 *bits* seguintes o peso da carga enviada ou o peso da classe presente no robô emissor. Os últimos 5 *bits* conterão o identificador do robô receptor da mensagem.

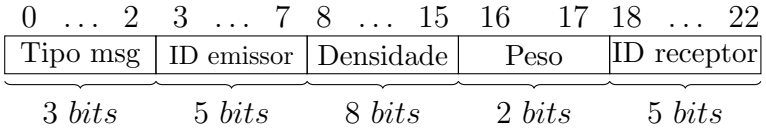


Figura 14: Estrutura da mensagem enviada pelo *Kilobot*

Durante a comunicação *broadcast local*, a informação é compartilhada unicamente com a vizinhança dentro do raio de comunicação. A mensagem enviada durante esta comunicação pelo robô i disponibiliza os valores de peso da classe u_i e a densidade de carga v_i. Note que a perda de informação durante esta comunicação não repercute na atualização do valor da densidade, pois neste caso o filtro de consenso deveria compensá-la. Por conseguinte, nesta comunicação não é preciso esperar uma confirmação. Note que para este tipo de comunicação não é preciso especificar o identificador do robô receptor, pois qualquer vizinho pode aproveitar a informação enviada para atualizar sua própria densidade de carga.

Durante a comunicação *ponto a ponto*, um robô i envia uma carga para um outro robô j, seguindo as fases detalhadas na Seção 3.1. O envio desta carga é fundamental para o funcionamento do algoritmo. Dado que o número de robôs no enxame possui um impacto direto no processo de comunicação, ocasionando um aumento no fluxo de mensagens trocadas, são necessárias duas confirmações da recepção da carga, para evitar qualquer perda. Na Figura 15 é apresentado o processo de troca de mensagens entre um robô emissor de uma carga e um robô receptor desta carga. A primeira confirmação tem a finalidade de prevenir a perda ou duplicação da carga enviada. A segunda confirmação tem a finalidade de prevenir a perda de uma segunda carga enviada consecutivamente pelo mesmo robô.

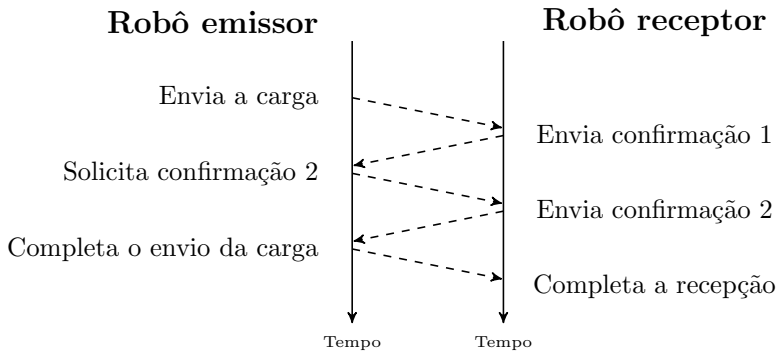


Figura 15: Troca de mensagens durante o envio de uma carga

4.3 Implementação

Para implementar o algoritmo de clusterização no *Kilobot* é utilizada uma máquina de estados de 5 estados. A Figura 16 resume o diagrama de transição de estados desta máquina, onde E_1 é o estado *Inicio*, E_2 o estado *Carregado*, E_3 o estado *Descarregado*, E_4 o estado *Carregado Dinâmico* e E_5 o estado Descarregado Dinâmico.

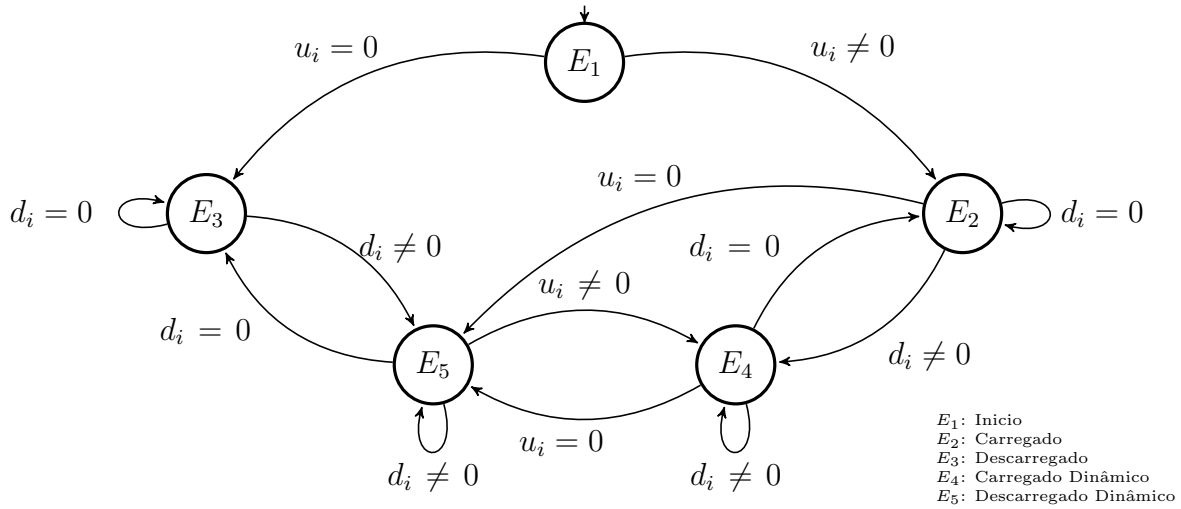


Figura 16: Máquina de estados do robô i

O estado E_1 representa o estado inicial do robô. Neste estado, o robô inicializa as variáveis e reconhece os robôs da sua vizinhança. A variável de densidade de carga v_i é inicializada com 0 e a variável u_i é inicializada randomicamente. Caso a variável u_i for inicializada com um valor positivo diferente de 0, o robô muda para o estado E_2, onde o comportamento do robô é carregado e não dispõe de cargas dinâmicas. Por conseguinte, o robô realiza a Fase 1, detalhada na Seção 3.1.1. Caso contrário, *i.e.* se a variável u_i

for inicializada com 0, o robô muda para o estado E_3, onde o comportamento do robô é descarregado sem cargas dinâmicas, comportamento explicado na Seção 3.2. Quando um robô estiver no estado E_2 e receber uma carga dinâmica, ele muda para o estado E_4. Neste estado como o robô possui uma carga estática, o robô realiza a Fase 1 para esta carga, posteriormente, realiza a fase correspondente à carga dinâmica. Finalmente, o estado E_5 trata unicamente as cargas dinâmicas. O comportamento do robô neste estado é descarregado dispondo de cargas dinâmicas.

Note que existem outras transições além das descritas. Caso o robô estiver no estado E_2 e carga estática transite para dinâmica, o robô muda para o estado E_5. A transição entre os estados E_3 e E_5 é determinada pela ausência de cargas dinâmicas. A transição entre os estados E_4 e E_5 é determinada pela variável u_i. Note que as transições são determinadas pela ausência de cargas dinâmicas, $d_i = 0$, ou pelo valor da variável u_i.

4.4 Métricas de avaliação

Como foi descrito no Capítulo 1, o algoritmo de agrupamento espacial poder ser expressado em termos das técnicas de clusterização como: *elemento*, *atributo*, *cluster*, *distância*, *similaridade* e *matriz de similaridade*. Nesta seção, são abstraídos os conceitos básicos da clusterização e relacionados com alguns dos resultados obtidos.

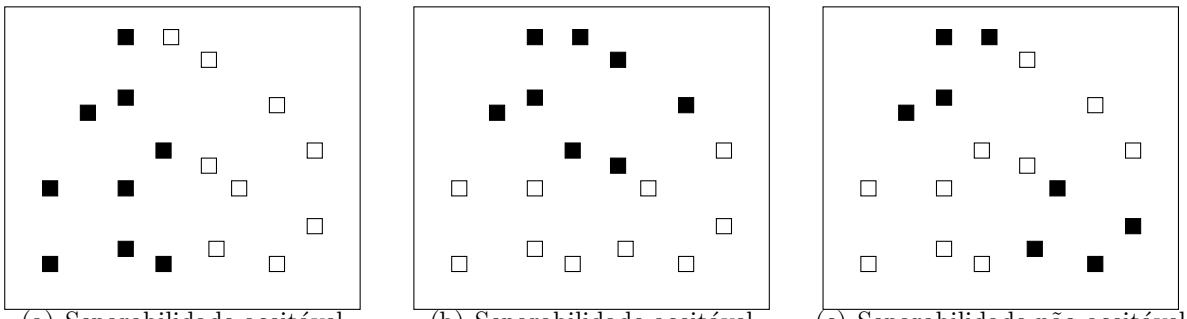
Os *elementos* a serem clusterizados, obviamente, são os robôs. Os robôs do enxame utilizam seus *atributos* de densidade de carga v_i e o peso da carga estática u_i para realizar a clusterização. Um *cluster* é definido como o grupo de robôs que são próximos, que apresentam o mesmo peso da carga estática e cujas densidades de carga estão dentro da mesma faixa de densidade, conforme definida na Equação 7, apresentada no Capítulo 3. A *similaridade* ou *distância* entre os elementos é determinada pela diferença entre as densidades das cargas dos robôs. Calculando a distância de cada um dos robôs com respeito aos demais robôs do enxame é possível elaborar a tabela de similaridade. A Equação 8 apresenta a estrutura da matriz de similaridade, onde V é a matriz de similaridade, e v_{ij} representa a medida de similaridade entre os robôs i e j. Note que quando $i = j$ a distância é 0.

$$V = \begin{pmatrix} 0 & v_{12} & \dots & v_{1j} & \dots & v_{1n} \\ v_{21} & 0 & \dots & v_{2j} & \dots & v_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ v_{i1} & v_{i2} & \dots & 0 & \dots & v_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ v_{n1} & v_{n2} & \dots & v_{nj} & \dots & 0 \end{pmatrix} \tag{8}$$

Também baseados nos conceitos de *separabilidade linear* (ELIZONDO, 2006) e *desequilíbrio de classes* (ROSAS et al., 2013), são analisados os resultados obtidos. Estas duas métricas são definidas a seguir.

4.4.1 Separabilidade Linear

Dois subgrupos X e Y são ditos linearmente separáveis se existir um hiperplano P tal que os elementos de X e Y estão em lados opostos do mesmo (ELIZONDO, 2006). Em outros termos, a *separabilidade linear* representa a possibilidade de visualizar os grupos individualmente, ou seja, a possibilidade de desenhar uma linha que separe os elementos do subgrupo. Nas Figuras 17 (a) e (b) são apresentados dois exemplos de conjuntos de pontos linearmente separáveis e na Figura 17 (c) um exemplo de um conjunto de pontos não-linearmente separáveis.



(a) Separabilidade aceitável (b) Separabilidade aceitável (c) Separabilidade não aceitável

Figura 17: Exemplo de separabilidade

A medida que avalia a separabilidade linear de 2 classes (DI CARO; DUCATELLE; GAMBARDELLA, 2012) é definida na Equação 9:

$$\sigma = \frac{Re}{n}, \tag{9}$$

onde σ é a medida separabilidade linear, Re é o número de robôs clusterizados de maneira errada e n o número total de robôs do enxame. Note que a separabilidade linear ótima é definida por $\sigma = 0$, ao passo que a pior separabilidade linear é indicada por $\sigma = 1$.

4.4.2 Desequilíbrio de Classes

Nos sistemas de aprendizado supervisionado é utilizado o conceito de *desequilíbrio de classes* (ROSAS et al., 2013). Em um conjunto de treinamento, é dito que existe um desequilíbrio de classes quando uma das classes é representada por uma grande quantidade de elementos, enquanto a outra é representada por poucos. Em outros termos, o desequilíbrio de classes avalia se o número de robôs nas 2 classes é balanceado.

A medida de desequilíbrio para um agrupamento em 2 classes (DI CARO; DUCATELLE; GAMBARDELLA, 2012) é definido por Equação 10:

$$\rho = \frac{Rm}{n}, \tag{10}$$

onde ρ é a medida de desequilíbrio entre as classes, Rm é o número de robôs da classe com menor número de elementos e, n é o número total de robôs do enxame. O melhor resultado do desequilíbrio é 0,5 e ocorre quando as duas classes apresentam o mesmo número de robôs, ao passo que o pior resultado é 0 e acontece quando uma das classes está vazia.

A métrica de desequilíbrio foi modificada nesta dissertação para abranger várias classes, considerando o número de classes desejado. Desta maneira, a métrica é normalizada para ficar entre 0 e 1, conforme a Equação 11:

$$\rho = \frac{Rm}{n}\zeta. \tag{11}$$

Note que para a métrica de separabilidade linear, conforme foi definida na Equação 9 (DI CARO; DUCATELLE; GAMBARDELLA, 2012), pode ser também utilizada para qualquer número de classes, não sendo exclusiva para 2 classes.

Visando visualizar a evolução das cargas estáticas durante o processo de clusterização, é realizado o cálculo de desequilíbrio de cargas estáticas. Para isso é utilizada a Equação 12:

$$\kappa = \frac{ce}{ce + cd}, \tag{12}$$

onde κ é o desequilíbrio de cargas, ce é o número de cargas estáticas, e cd é o número de cargas dinâmicas presentes no enxame.

4.5 Resultados Obtidos

Foram realizadas 63 experiências mudando o número de classes, o número de robôs no enxame e a formação dos robôs nos cenários experimentados. A totalidade dos resultados obtidos está disponível no Apêndice A. Nesta seção são comentadas algumas destas experiências. Também, serão apresentadas análises do impacto do número de robôs, do número de cargas estáticas iniciais e do número de classes em relação ao tempo de clusterização.

Para facilitar a visualização da clusterização, foi programado o LED do robô para representar o peso da classe e a movimentação das cargas dinâmicas pelo robô. A Tabela 2 apresenta a configuração de cores utilizadas na implementação.

Tabela 2: Cores do LED do robô utilizadas na implementação

Classe	Sem cargas dinâmicas	Com cargas dinâmicas[1]
0	Azul	Ciano
1	Vermelho	Amarelo
2	Magenta	Branco
3	Verde	Desligado

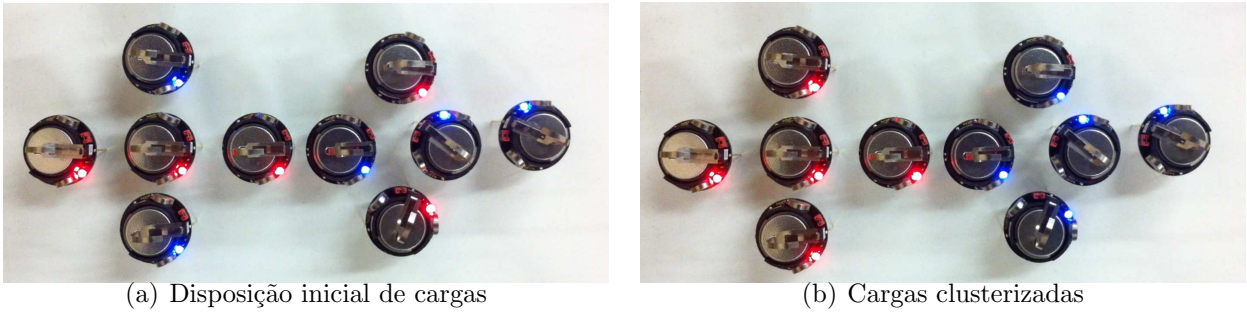
4.5.1 Experiências com 2 classes

Considerando 2 classes, foram realizadas 23 experiências, utilizando disposições diferentes em enxames de 9 até 28 robôs. A Figura 18 apresenta um enxame de 10 robôs, 2 classes e com 5 cargas estáticas iniciais. A Figura 19 apresenta a evolução da resposta do sistema. Inicialmente, o sistema proporciona uma separabilidade linear de 0,4, desequilíbrio de classes de 1 e desequilíbrio de cargas de 1. Note que, para 2 classes, se $ce + cd = \frac{n}{2}$ o valor do desequilíbrio de classes e do desequilíbrio de cargas são iguais, já que, para 2 classes $Rm = ce$. Das Equações 11 e 12 obtemos a Equação 13:

$$\kappa = \frac{2Rm}{n}. \tag{13}$$

O algoritmo converge em $64\,s$, com uma separabilidade linear ótima de 0, possibilitando a distinção dos 2 grupos de robôs, e um desequilíbrio de classes de 1, o que proporciona uma disposição final de classes balanceada, ou seja, as duas classes têm o mesmo número de robôs. A Figura 18 (b) apresenta o resultado final da clusterização.

[1]Essas cores somente serão visualizadas no vídeo.



(a) Disposição inicial de cargas (b) Cargas clusterizadas

Figura 18: Enxame de 10 *Kilobots*, 2 classes e 5 cargas estáticas iniciais

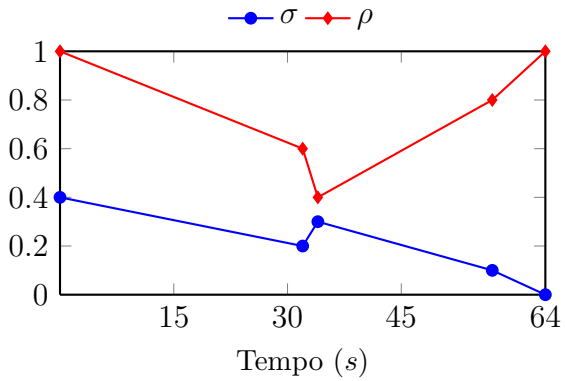
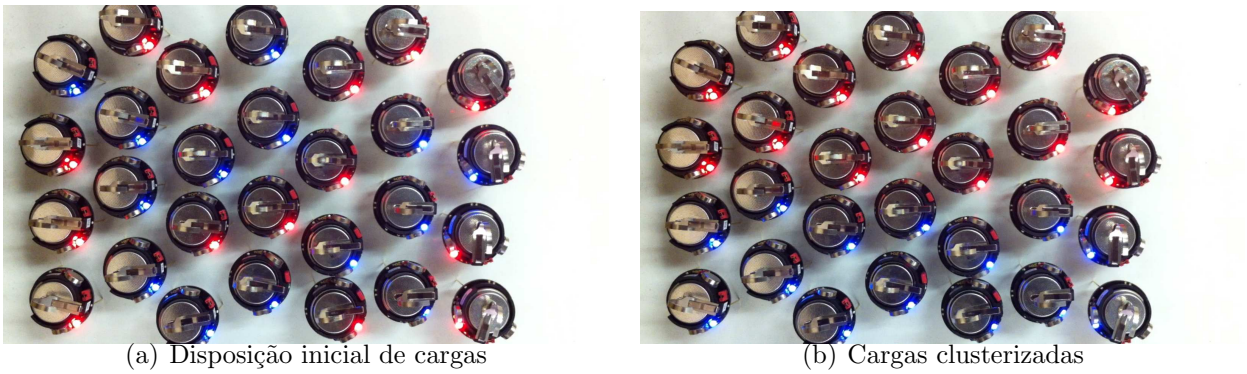


Figura 19: Evolução temporal da separabilidade linear e o desequilíbrio de um enxame de 10 *Kilobots*, 2 classes e 5 cargas estáticas iniciais

A Figura 20 apresenta um enxame de 28 robôs, 2 classes e 14 cargas estáticas iniciais. A Figura 21 apresenta a evolução da resposta do sistema. Inicialmente, o sistema apresenta uma separabilidade linear de 0,5 e desequilíbrio de classes de 1.



(a) Disposição inicial de cargas (b) Cargas clusterizadas

Figura 20: Enxame de 28 *Kilobots*, 2 classes e 14 cargas estáticas iniciais

O algoritmo converge em 242 s, com uma separabilidade linear de 0 e um desequilíbrio de classes de 1. A Figura 20 (b) apresenta o resultado final da clusterização.

Foram usados enxames de diferentes tamanhos, variando entre 9 e 28 o número de robôs. A Tabela 3, mostrada no Anexo A, apresenta as principais características das

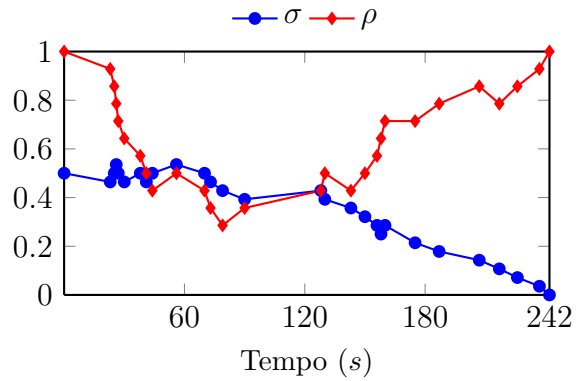


Figura 21: Evolução temporal da separabilidade linear e o desequilíbrio de um enxame de 28 *Kilobots*, 2 classes e 14 cargas estáticas iniciais

23 experiências realizadas para 2 classes. Os dados foram obtidos depois de terminada a clusterização. No Anexo A, são apresentadas as disposições utilizadas em cada uma das experiências.

Para analisar o impacto do número de robôs do enxame no tempo de convergência em uma clusterização de 2 classes, foram realizadas 9 experiências fixando em 5 o número de cargas estáticas iniciais. A Figura 22 apresenta a relação entre o tempo de clusterização e o número de robôs.

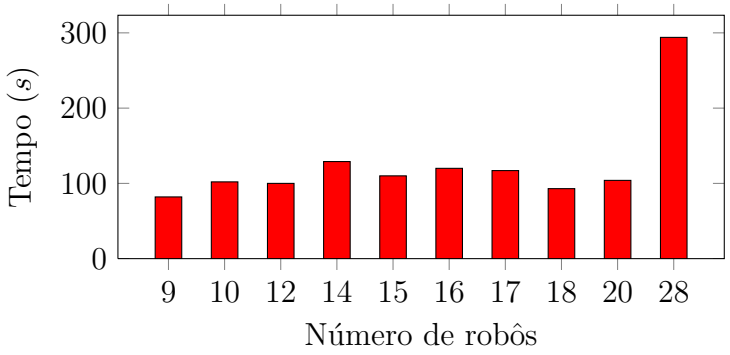
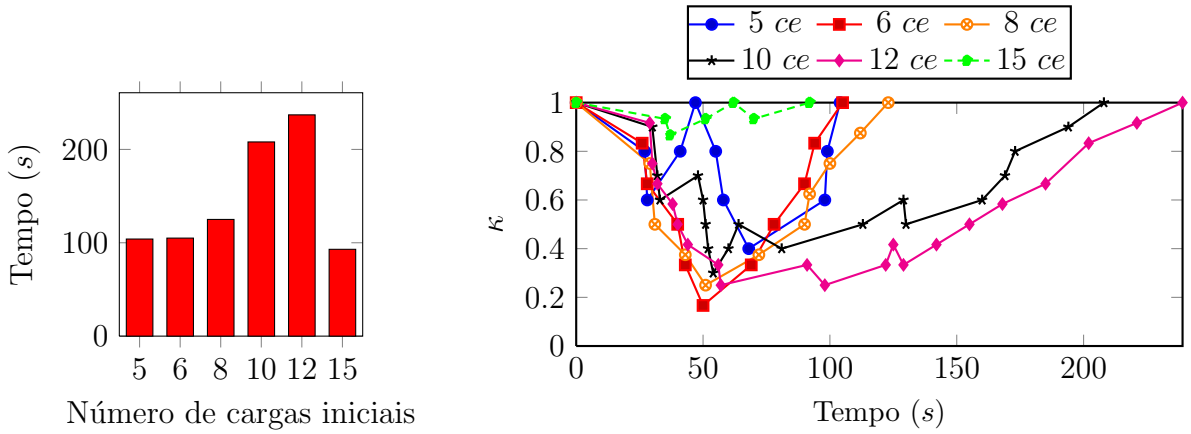


Figura 22: Relação entre o número de robôs e o tempo de clusterização em 2 classes

Os resultados mostram um aumento no tempo de convergência. Embora não seja significativa a diferença para as experiências de 9 a 20 robôs, a experiência com 28 robôs quase triplica o tempo de clusterização. A pouca diferença de tempo entre os enxames de 9 a 20 robôs é explicada pelo número de cargas existentes no sistema em relação ao tamanho do enxame. As 5 cargas conseguem gerar um alto valor de densidade de cargas estáticas em pouco tempo, facilitando a geração de um *cluster* inicial que guia as outras

cargas ainda espalhadas. Entretanto, no enxame com 28 robôs, o que aumenta o número de vizinhos de cada robô, as cargas não conseguem produzir áreas muito densas desde o início.

Para analisar o impacto do número de cargas no tempo de clusterização, foram realizadas 6 experiências com o número de robôs do enxame fixado em 20. Esta relação é apresentada na Figura 23 (a).



(a) Relação entre cargas e tempo

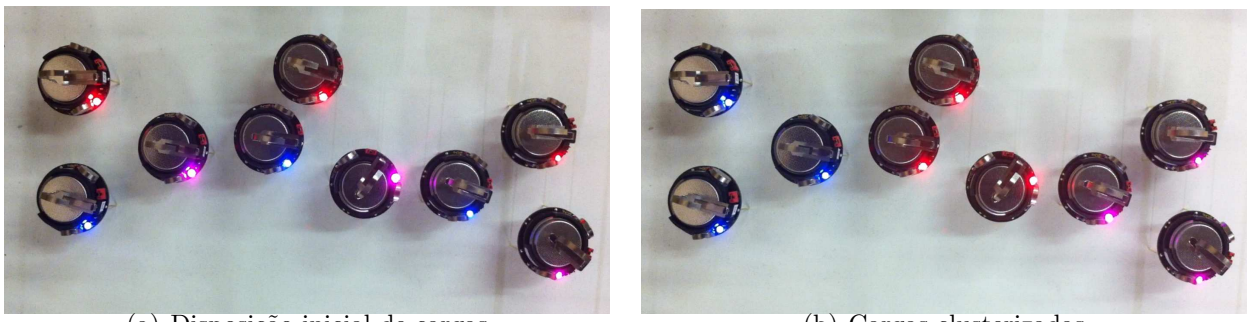
(b) Evolução do desequilíbrio de cargas

Figura 23: Relação entre o número de cargas e o tempo de clusterização em 2 classes

A Figura 23 (b) apresenta a evolução do desequilíbrio de cargas. Nos resultados obtidos foi evidenciado que o tempo de convergência diminui quando existem poucas cargas dinâmicas ao mesmo tempo. Isto acontece nas experiências com 5, 6 e 8 cargas estáticas iniciais, pois o número máximo de cargas dinâmicas durante a execução foi 3, 6 e 5 respectivamente. Ao realizar poucas trocas de cargas, o número de colisões das mensagens transmitidas é menor. No caso de 10 e 12 cargas estáticas iniciais, o tempo de clusterização foi maior devido à movimentação de cargas durante a execução do algoritmo. Nestes dois casos o número máximo de cargas dinâmicas durante a execução foi de 7 e 9 respectivamente. Também, para uma maior quantidade de cargas, foi evidenciado que o aumento de robôs carregados na inicialização impede a correta execução do algoritmo. Esta limitação do algoritmo é devido ao aumento acelerado da densidade de cargas estáticas em todos os robôs do enxame. Apesar disso, para 15 cargas estáticas iniciais, foi possível a movimentação de 3 cargas durante a execução da experiência. Este é o motivo pelo qual o tempo de clusterização é relativamente baixo.

4.5.2 Experiências com 3 classes

Considerando 3 classes, foram realizadas 22 experiências, utilizando disposições diferentes em enxames de 9 até 28 robôs. A Figura 24 apresenta um enxame de 9 robôs, 3 classes e 6 cargas estáticas iniciais, sendo 3 de classe 1 e 3 de classe 2. A Figura 25 apresenta a evolução da resposta do sistema. Inicialmente, o sistema proporciona uma separabilidade linear de 0,67, desequilíbrio de classes de 1 e desequilíbrio de cargas de 1.



(a) Disposição inicial de cargas (b) Cargas clusterizadas

Figura 24: Enxame de 9 *Kilobots*, 3 classes e 6 cargas estáticas iniciais

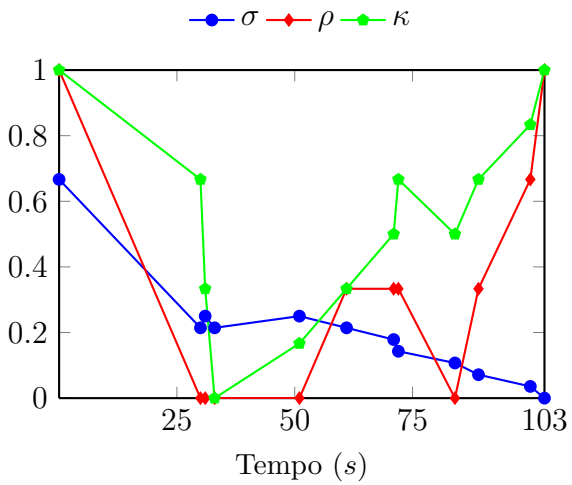


Figura 25: Evolução temporal da separabilidade linear e o desequilíbrio de um enxame de 9 *Kilobots*, 3 classes e 6 cargas estáticas iniciais

O algoritmo converge em 103 s, com uma separabilidade linear de 0, o desequilíbrio de classes 1 significando que a disposição final de classes é balanceada, e o desequilíbrio de cargas igual a 1 levando a uma proporção de cargas balanceada, não tendo perda de cargas. A Figura 24 (b) apresenta o resultado final da clusterização.

A Figura 26 apresenta um enxame de 18 robôs, 3 classes e 12 cargas estáticas iniciais, sendo 6 de classe 1 e 6 de classe 2. A Figura 27 apresenta a evolução da resposta do

sistema. Inicialmente, o sistema apresenta uma separabilidade linear de 0,5 e desequilíbrio de classes de 1.

(a) Disposição inicial de cargas (b) Cargas clusterizadas

Figura 26: Enxame de 18 *Kilobots*, 3 classes e 12 cargas estáticas iniciais

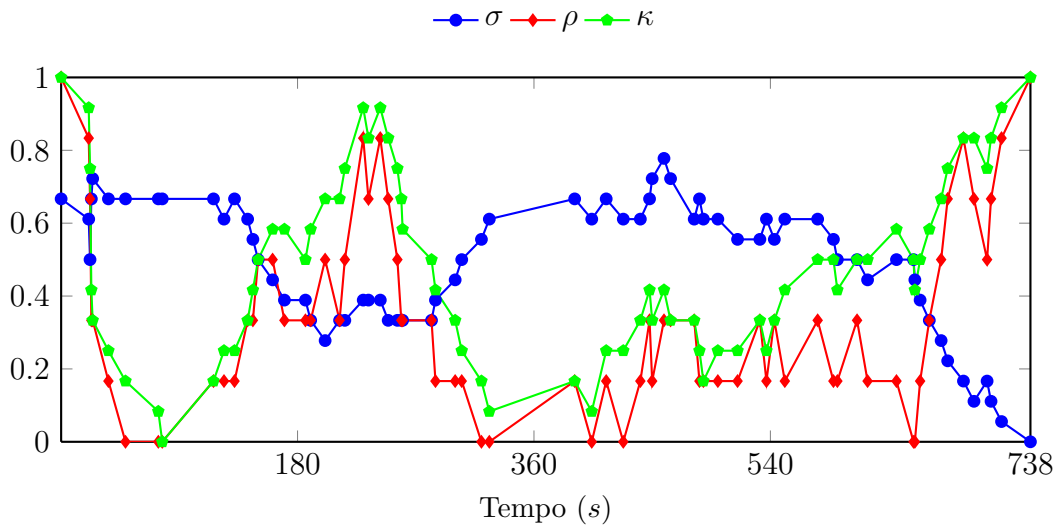


Figura 27: Evolução temporal da separabilidade linear e o desequilíbrio de um enxame de 18 *Kilobots*, 3 classes e 12 cargas estáticas iniciais

O algoritmo converge em 738 s, com uma separabilidade linear de 0, o que significa que é possível distinguir os 3 grupos de robôs, o desequilíbrio de classes 1 sendo que a disposição final de classes é balanceada, e o desequilíbrio de cargas de 1 sendo que a proporção de cargas é também balanceada. Não houve perda de cargas. A Figura 26 (b) apresenta o resultado final da clusterização.

A Tabela 4, mostrada no Anexo A, apresenta as principais características das 22 experiências realizadas para 3 classes. Foram usados enxames de diferentes tamanhos, variando entre 9 e 28 o número de robôs. Os dados foram obtidos depois de terminada a

clusterização. No Anexo A, são apresentadas as disposições utilizadas em cada uma das experiências, assim como os gráficos de evolução no tempo.

Foi analisado o impacto do tamanho do enxame no tempo de convergência em uma clusterização de 3 classes. Foram realizadas 9 experiências fixando em 6 o número de cargas estáticas iniciais, sendo 3 cargas de classe 1 e 3 de classe 2. A Figura 28 apresenta a relação entre o tempo de clusterização e o número de robôs.

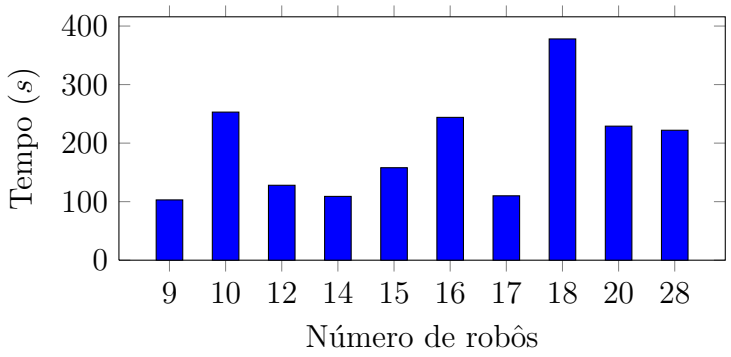
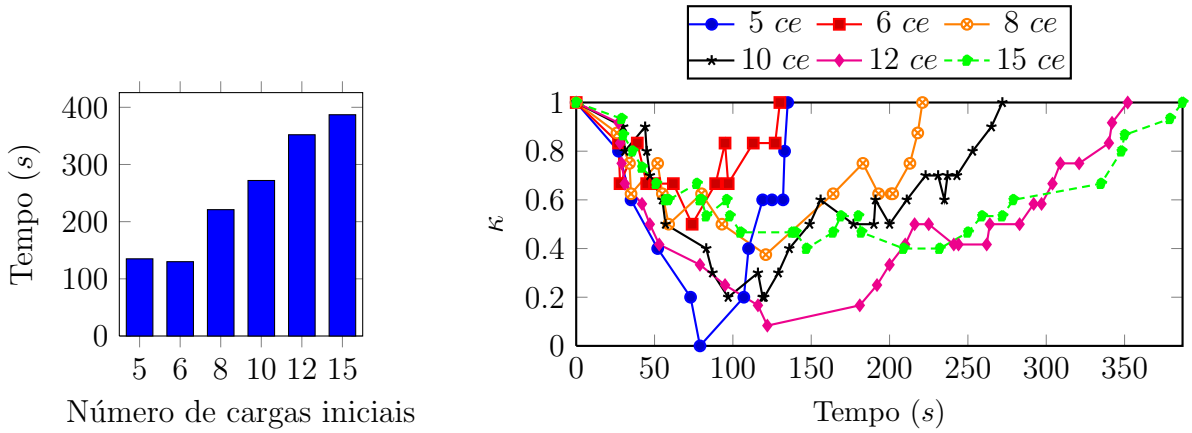


Figura 28: Relação entre o número de robôs e o tempo de clusterização em 3 classes

Nas experiências representadas na Figura 28, é observado que o tempo de clusterização é levemente influenciado pelo número de robôs. No caso da experiência com 18 robôs, cujo tempo de clusterização foi de 378 s, é evidenciado que, para certas formações do enxame, podem acontecer gargalos na comunicação entre os robôs, introduzindo um atraso no envio de cargas. Entretanto, na experiência com 17 robôs, com tempo de clusterização de 110 s, é evidenciado que, se um robô carregado está posicionado adequadamente, este pode facilitar a clusterização.

Para analisar o impacto do número de cargas no tempo de clusterização, foram realizadas 6 experiências com o número de robôs do enxame fixado em 20. Esta relação é apresentada na Figura 29 (a).

A Figura 29 (b) apresenta a evolução do desequilíbrio de cargas. Similarmente as experiências realizadas com 2 classes, foi evidenciado que o aumento das cargas estáticas iniciais no sistema, introduz um aumento no tempo de convergência. Também, foi observado que as cargas com maior peso facilitam a clusterização das cargas com menor peso. O aumento do número de robôs carregados na inicialização não dificulta a correta clusterização. Embora a densidade aumente rapidamente, a existência de dois tipos de cargas diferentes gera faixas de densidade, permitindo uma movimentação errada das cargas clusterizadas.



(a) Relação entre cargas e tempo

(b) Evolução do desequilíbrio de cargas

Figura 29: Relação entre o número de cargas e o tempo de clusterização em 3 classes

4.5.3 Experiências com 4 classes

Considerando 4 classes, foram realizadas 18 experiências, utilizando disposições diferentes em enxames de 16 até 28 robôs. A Figura 30 apresenta um enxame de 20 robôs, 4 classes e 15 cargas estáticas iniciais, sendo 5 de classe 1, 5 de classe 2 e 5 de classe 3.

A Figura 31 apresenta a evolução da resposta do sistema. Inicialmente, o sistema apresenta uma separabilidade linear de 0,67, um desequilíbrio de classes de 1 e desequilíbrio de cargas de 1.

(a) Disposição inicial de cargas

(b) Cargas clusterizadas

Figura 30: Enxame de 20 *Kilobots*, 4 classes e 15 cargas estáticas iniciais

O algoritmo converge em 103 s, com uma separabilidade linear de 0, o desequilíbrio de classes de 1 e um desequilíbrio de cargas de 1. Não houve perda de cargas. A Figura 30 (b) apresenta o resultado final da clusterização.

A Tabela 5, mostrada no Apêndice A, apresenta as principais características das 18 experiências realizadas para 4 classes. Nas experiências, o tamanho do enxame varia

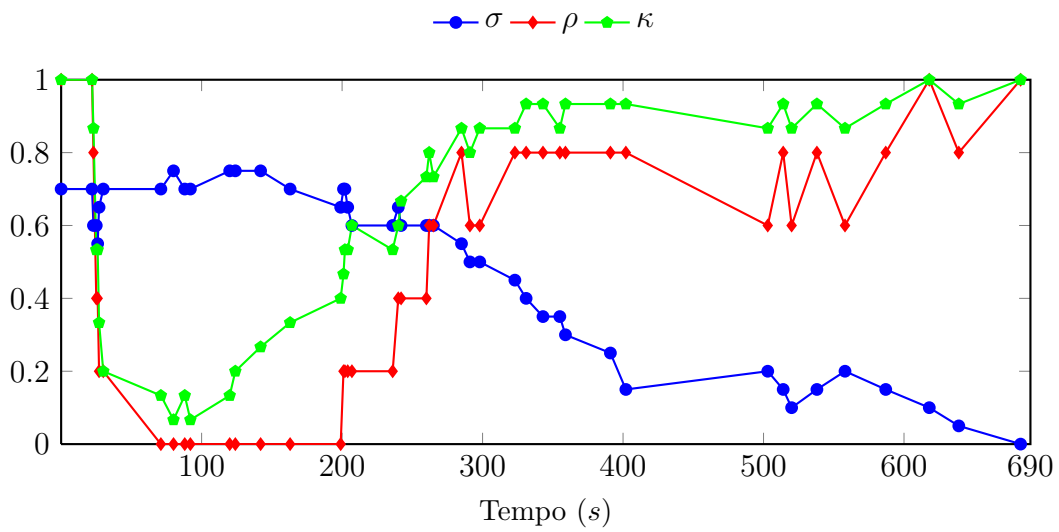


Figura 31: Evolução temporal da separabilidade linear e o desequilíbrio de um enxame de 20 *Kilobots*, 4 classes e 15 cargas estáticas iniciais

entre 9 e 28 robôs. Os dados foram obtidos depois de terminada a clusterização. No Anexo A, são apresentadas as disposições utilizadas em cada uma das experiências.

Foi analisado o impacto do tamanho do enxame no tempo de convergência em uma clusterização de 4 classes. Foram realizadas 9 experiências fixando em 8 o número de cargas estáticas iniciais, sendo 2 cargas de classe 1, 3 cargas de classe 2 e 3 cargas de classe 3. A Figura 32 apresenta a relação entre o tempo de clusterização e o número de robôs.

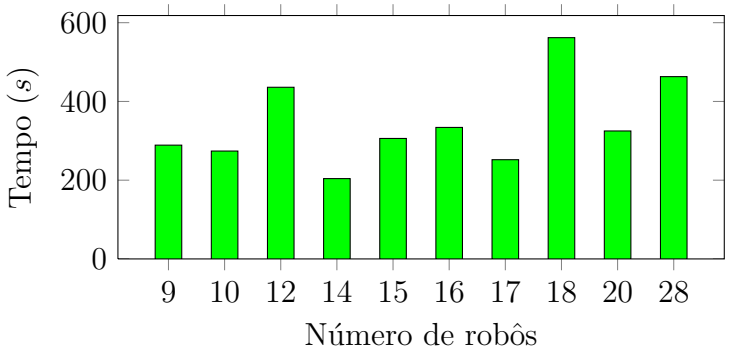
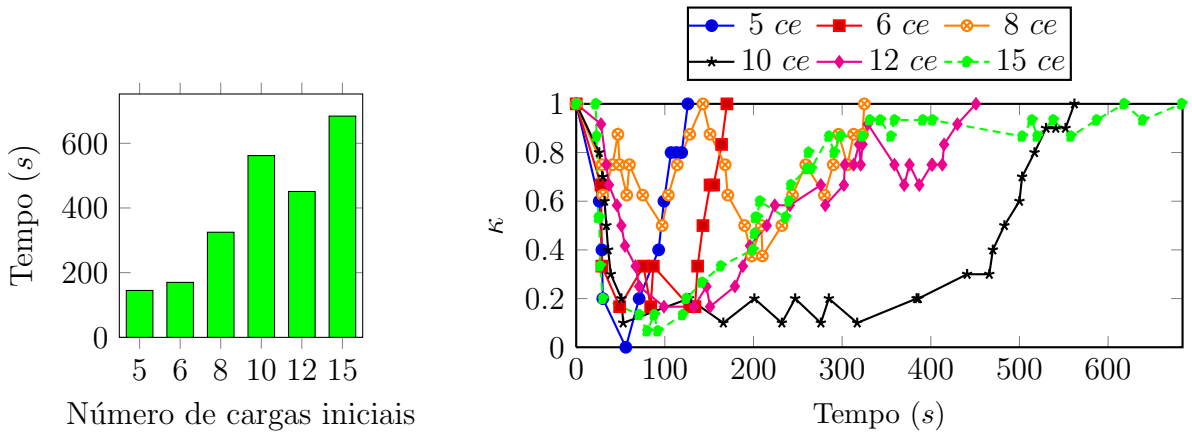


Figura 32: Relação entre o número de robôs e o tempo de clusterização em 4 classes

Os resultados obtidos mostraram novamente que o aumento no número de robôs aumenta levemente o tempo de clusterização. Também, foi observado que a disposição do enxame é um fator relevante para o processo de clusterização. Quando várias cargas dinâmicas são guiadas ao mesmo tempo para um robô com alta densidade, este não consegue atender todas as demandas, formando um gargalo durante a comunicação. Isto

afeta diretamente ao tempo de convergência.

Para analisar o impacto do número de cargas no tempo de clusterização, foram realizadas 6 experiências com o número de robôs do enxame fixado em 20. Esta relação é apresentada na Figura 33 (a).



(a) Relação entre cargas e tempo

(b) Evolução do desequilíbrio de cargas

Figura 33: Relação entre o número de cargas e o tempo de clusterização em 4 classes

A Figura 33 (b) apresenta a evolução do desequilíbrio de cargas. Como esperado, ao aumentar o número de cargas estáticas iniciais, o tempo de clusterização aumento proporcionalmente. Não obstante, foi observado que o incremento de cargas de maior peso facilita a clusterização das demais cargas. Isto acontece devido ao aumento da densidade de cargas estáticas, que guia as cargas dinâmicas.

Para complementar as análises dos resultados levantados, foram comparados os tempos de clusterização para sistemas de 2, 3 e 4 classes. A Figura 34 apresenta os resultados obtidos variando o número de robôs e fixando em 8 o número de cargas estáticas iniciais. Para a clusterização em 2 classes, todas as cargas foram tipo 1. Para a clusterização em 3 classes, 4 cargas foram tipo 1 e 4 tipo 2. Para a clusterização em 3 classes, 2 cargas foram tipo 1, 3 tipo 2 e 3 tipo 3.

Na Figura 35, são comparados os resultados variando o número de cargas estáticas iniciais e mantendo fixo o tamanho do enxame em 20 robôs.

As Figuras 34 e 35 apresentam resultados esperados, evidenciando que o tempo de clusterização aumenta com o número de classes no sistema. Também, é possível observar que o número de cargas no sistema afeta negativamente o tempo de convergência. Como foi explicado anteriormente, se o sistema é inicializado com muitos robôs carregados, estes incrementaram rapidamente o valor da densidade de carga estática, impossibilitando a

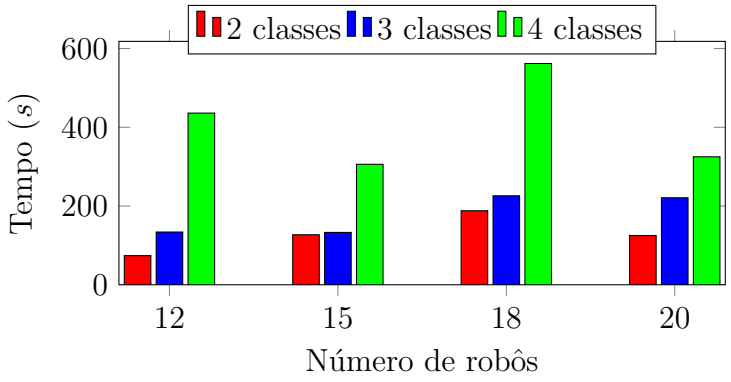


Figura 34: Relação entre o número de robôs e o tempo de clusterização

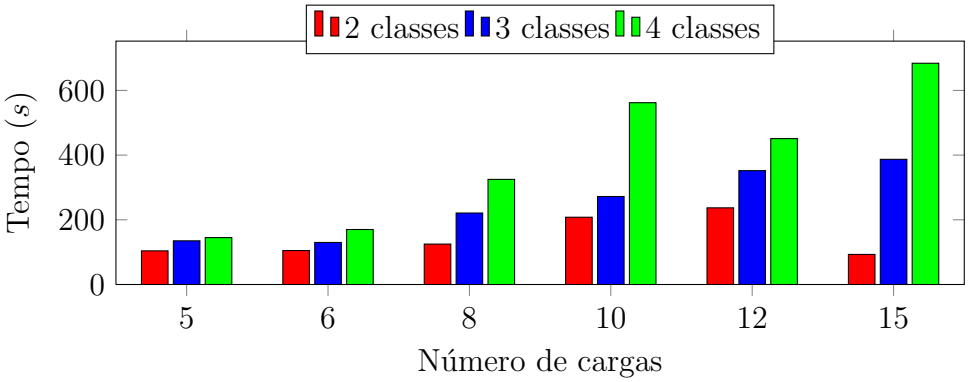


Figura 35: Relação entre o número de cargas e o tempo de clusterização

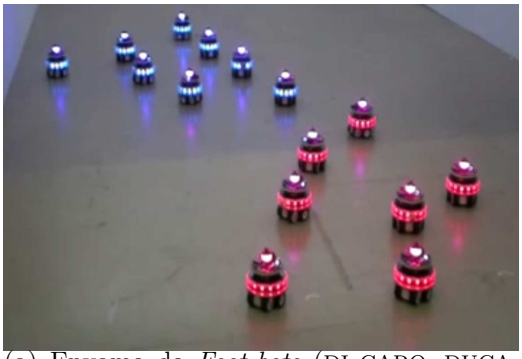
movimentação das cargas e por consequência impedindo a correta clusterização. Os vídeos das experiências realizadas estão disponíveis através do *link*: `http://bit.ly/1mVxBci`.

4.5.4 Comparação

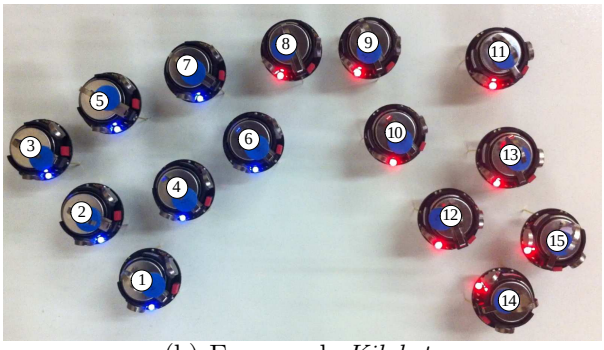
Visando ter uma ponto de referência nos resultados obtidos foi realizada uma comparação com os resultados publicados em (DI CARO; DUCATELLE; GAMBARDELLA, 2012) utilizando robôs reais do tipo *Foot-boot*. Os *Foot-boot* possuem um microprocessador iMX31 de 532 MHz. Esta comparação não visava obter melhores resultados, mas sim ter uma validação da resposta do sistema, assim como o tempo de referência para clusterizar. Todas as experiências realizadas em (DI CARO; DUCATELLE; GAMBARDELLA, 2012) foram unicamente com 2 classes.

Para esta experiência foram utilizados 15 robôs e 8 cargas estáticas distribuídas aleatoriamente no enxame. A Figura 36 (a) apresenta a formação do enxame de *Foot-boots* utilizada em (DI CARO; DUCATELLE; GAMBARDELLA, 2012), e a Figura 36 (b) apresenta a formação do enxame de *Kilobots*, observe que a formação é a mesma nas duas experi-

ências. A evolução da resposta do sistema é apresentada na Figura 37. A convergência da clusterização acontece em 120 s no caso do enxame de *Footbots* e em 135 s no caso do enxame de *Kilobots*.



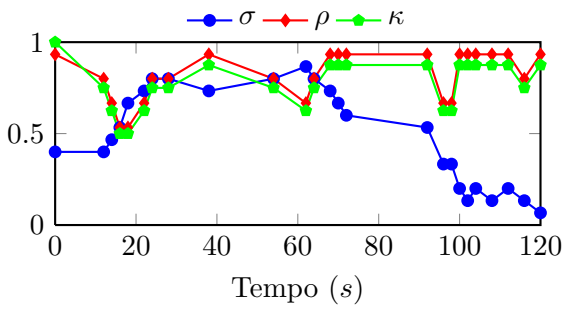
(a) Enxame de *Foot-bots* (DI CARO; DUCATELLE; GAMBARDELLA, 2012)



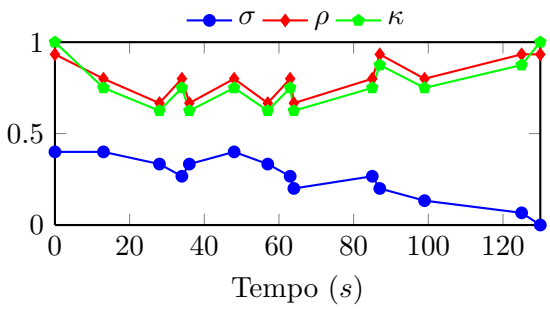
(b) Enxame de *Kilobots*

Figura 36: Formação utilizada na primeira experiência

Em $t = 0\,s$, a separabilidade linear é de 0,4, o desequilíbrio de classes é de 0,933 e o desequilíbrio de cargas de 1 em ambos os sistemas. Em $t = 75\,s$, o enxame de *Foot-bots* apresenta uma separabilidade linear de 0,6, desequilíbrio de classes de 0,9333 e desequilíbrio de cargas de 0,875. Por outro lado, o enxame de *Kilobots* apresenta uma separabilidade linear de 0,2, desequilíbrio de classes de 0,666 e desequilíbrio de cargas de 0,625. Em $t = 125\,s$, os *Foot-bots* convergem para uma solução, com uma separabilidade linear de 0,066, um desequilíbrio de classes de 0,933 e desequilíbrio de cargas de 0,875. Note que, neste caso a separabilidade linear não é igual a 0, isto acontece devido à perda de uma carga, e só clusterizaram 7 cargas das 8 iniciais. Em contraste, o enxame de *Kilobots* converge em $t = 130\,s$, mas com uma separabilidade linear de 0, um desequilíbrio de



(a) Resposta do enxame de *Footbots*



(b) Resposta do enxame de *Kilobots*

Figura 37: Evolução temporal da separabilidade linear e do desequilíbrio na primeira experiência

classes de 0,933 e um desequilíbrio de cargas de 1. Note que nos dois casos o desequilíbrio de classes não é igual a 1, isto é devido ao fato que o número de robôs do enxame é impar, então a proporção das classes nunca será igual. O desequilíbrio de cargas permite ressaltar a perda da carga no dos *Foot-bots*.

4.6 Considerações Finais do Capítulo

Neste capítulo foi apresentada a implementação do algoritmo de clusterização espacial usando robôs do tipo *Kilobot*. Foram descritos os conceitos de *separabilidade linear* e *desequilíbrio de classes*, que são as métricas utilizadas na avaliação da implementação. Foram apresentados alguns dos resultados obtidos com a implementação para 2, 3 e 4 classes, com diferentes disposições dos robôs e distintos tamanhos do enxame. Também foi realizada a comparação dos resultados obtidos com os publicados em (DI CARO; DUCATELLE; GAMBARDELLA, 2012).

O capítulo seguinte finaliza este trabalho, abordando suas principais conclusões, bem como os pontos mais relevantes da presente dissertação. Também serão tratadas direções para possíveis trabalhos futuros.

Capítulo 5

CONCLUSÕES E TRABALHOS FUTUROS

NO decorrer desta dissertação, foi proposto, implementado e analisado um algoritmo distribuído de clusterização espacial de robôs. Baseado na ideia de clusterização para 2 classes publicado em (DI CARO; DUCATELLE; GAMBARDELLA, 2012), o algoritmo proposto consegue clusterizar um enxame de robôs com características homogêneas em 2 ou mais classes. Este capítulo apresenta as principais conclusões deste trabalho, assim como indicações de possíveis trabalhos futuros.

5.1 Conclusões

Esta dissertação abordou a clusterização espacial em enxames de robôs. A realização deste trabalho é incentivada pela necessidade de estudar alternativas na alocação de tarefas. A possibilidade de trabalhar cooperativamente entre grandes grupos de agentes ou indivíduos acarreta a necessidade de uma coordenação das atividades a serem realizadas. De forma geral, a clusterização espacial de sistemas distribuídos permite alocar tarefas para robôs localizados em áreas próximas, facilitando a cooperação.

O algoritmo proposto permite dividir em classes um enxame de robôs, sem precisar da movimentação deles nem precisar de informação global sobre o sistema. Para implementar o algoritmo de clusterização espacial foram utilizados robôs do tipo *Kilobot*, projetado para estudar teorias de Inteligência de Enxame e comportamentos coletivos. Nas experiências realizadas nos *Kilobots* foram testados casos com até 4 subgrupos, sendo possível implementar qualquer número de subgrupos. A clusterização de robôs é realizada por meio de distribuição de cargas virtuais. Quando uma carga fica estática em um robô, é calculada uma densidade de carga. Esta densidade é proporcional ao número de robôs

com cargas estáticas na vizinhança. Com a informação de densidade compartilhada de maneira local, os robôs dirigem as cargas dinâmicas para áreas com alta densidade.

Foram realizadas 63 experiências com configurações diferentes. Foi representada a evolução temporal do sistema utilizando métricas sobre a separabilidade linear, o desequilíbrio de classes e o desequilíbrio de cargas estáticas. Os resultados obtidos foram analisados considerando o tempo de convergência e a relação com o número de robôs do enxame, de cargas estáticas iniciais e de classes da clusterização. Os resultados obtidos demostraram uma relação direta entre o número de cargas dinâmicas e o tempo de convergência. Este resultado é explicado pela relação direta entre o número de cargas dinâmicas e o volume de troca de mensagens. Também, foi evidenciado que o aumento do número de robôs do enxame afeta levemente o tempo de clusterização. No entanto, a distribuição dos robôs no enxame pode afetar o processo de troca de mensagens e aumentar o tempo de convergência.

Nas experiências realizadas foi observado que, o tempo de convergência é afetado negativamente pelo número de classes da clusterização. As faixas de densidade, geradas pelos diferentes tipos de cargas, podem retardar a convergência do processo de clusterização. Quando um robô, carregado corretamente não consegue ficar dentro da faixa de densidade correta, a carga estática dele tende a se movimentar, o que leva a um atraso no processo de clusterização.

O algoritmo apresenta restrições relacionadas à quantidade de cargas estáticas iniciais em relação ao tamanho do enxame. Isto foi observado para a clusterização em 2 classes. Se na inicialização do sistema existe uma grande quantidade de robôs carregados em relação à quantidade de robôs descarregados, a densidade de cargas estáticas terá um aumento acelerado, impedindo a correta convergência do processo de clusterização. O aumento acelerado da densidade impossibilita a movimentação das cargas. Por outro lado, se durante a inicialização do sistema a quantidade de robôs carregados é mínima em relação à quantidade de robôs descarregados, a densidade de cargas estáticas não é mantida dentro da faixa certa, ocasionando que o cluster formado não fique estável.

A implementação realizada nos *Kilobots* foi comparada com os resultados publicados em (DI CARO; DUCATELLE; GAMBARDELLA, 2012). O algoritmo foi comparado com uma clusterização em 2 classes, implementado em robôs *Foot-bots*. Os tempos de convergência foram similares, tendo só uma diferença de $5s$. Esta diferença é muito pequena

considerando que o microprocessador do *Foot-bot* é mais robusto e possui uma frequência de processamento de 532 MHz, enquanto o microprocessador do *Kilobot* possui uma frequência de processamento de 8 MHz. Por não possuir um sistema eficaz de confirmação de envio de carga, o sistema de *Foot-bots* pode perder cargas durante o processo de clusterização. Enquanto, o sistema implementado nos *Kilobots* não apresentou este problema nas experiências realizadas.

5.2 Trabalhos Futuros

Nesta seção, são citadas algumas possíveis melhorias no algoritmo proposto com intuito de melhorar o seu desempenho. Também é incentivada a realização de novos ensaios que busquem avaliar diferentes aspectos do algoritmo proposto, tais como: o impacto da quantidade de vizinhos em cada robô e o consumo energético no processo de clusterização.

É incentivada a implementação do algoritmo de clusterização espacial em robôs que utilizem um sistema de comunicação diferente a infravermelho. Mesmo utilizando CSMA/CA (*Carrier sense multiple access with collision avoidance*) como método para a redução da ocorrência de colisões, foi observado a perda de informação durante as experiências com maior número de troca de mensagens. Utilizando outro sistema de comunicação como RF (Radio Frequência), é possível evitar a perda de informação.

Para melhorar o tempo de clusterização é recomendado aperfeiçoar a troca de mensagens entre os robôs, além do uso de outro sistema de comunicação. O uso de duas confirmações para envio de cargas dinâmicas, usando comunicação infravermelha, satura o canal de transmissão. Isto leva a aumentar o tempo de transmissão das mensagens.

A implementação de um robô líder junto com o algoritmo de clusterização espacial permitiria melhorar o algoritmo. A eleição de um robô líder permitiria resolver o problema da baixa densidade de carga estática, fixando um ponto de referência para o começo da clusterização. Além disso, para solucionar o problema da alta densidade inicial, poderia ser utilizada uma mensagem de retroalimentação verificando o resultado da clusterização.

REFERÊNCIAS

ABBASI, A. A.; YOUNIS, M. A survey on clustering algorithms for wireless sensor networks. *Computer communications*, Elsevier, v. 30, n. 14, p. 2826–2841, 2007.

ARAÚJO, R. N. *Análise de clusterização: Estudo comparativo de mecanismos de agrupamento usando o algoritmo de colônia de formigas*. Recife: Escola Politécnica de Pernambuco, Universidade de Pernambuco, 2012.

ATMEL. *ATmega328p*. USA, 2013. Disponível em: <http://www.atmel.com>.

BASU, P.; KHAN, N.; LITTLE, T. D. A mobility based metric for clustering in mobile ad hoc networks. In: IEEE. *Distributed Computing Systems Workshop, 2001 International Conference on*. USA, 2001. p. 413–418.

BECKERS, R.; HOLLAND, O.; DENEUBOURG, J.-L. From local actions to global tasks: Stigmergy and collective robotics. In: *Artificial life IV*. United States of America: the Massachusetts Institute of Technology, 1994. v. 181, p. 189.

BENI, G.; WANG, J. Swarm intelligence in cellular robotic systems. In: *Robots and Biological Systems: Towards a New Bionics?* California, USA: Springer, 1993. p. 703–712.

BONABEAU, E.; DORIGO, M.; THERAULAZ, G. *Swarm intelligence: from natural to artificial systems*. New York, USA: Oxford university press, 1999. ISBN 0-19-513158-4.

BULLA, N.; NEDJAH, N.; MOURELLE, L. Agrupamento espacial eficiente em $\zeta \geq 2$ classes para robótica de enxame. In: CBA. *Congresso Brasileiro de Automática. CBA 2014*. Belo Horizonte, 2014.

BULLA, N.; NEDJAH, N.; MOURELLE, L. Agrupamiento espacial eficiente en $\zeta \geq 2$ clases para robótica de enjambre. In: IEEE. *Congreso Bienal de IEEE Argentina. ARGENCON 2014*. San Carlos de Bariloche, Argentina, 2014.

CHEN, J. et al. Segregation in swarms of e-puck robots based on the brazil nut effect. In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2012. (AAMAS '12), p. 163–170.

DI CARO, G. A.; DUCATELLE, F.; GAMBARDELLA, L. A fully distributed communication-based approach for spatial clustering in robotic swarms. In: *Proceedings of the 2nd Autonomous Robots and Multirobot Systems Workshop (ARMS), affiliated with the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. Valencia, Spain, June 5: IFAAMAS, 2012. p. 153–171.

DORIGO, M.; MANIEZZO, V.; COLORNI, A. Ant system: optimization by a colony of cooperating agents. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, IEEE, v. 26, n. 1, p. 29–41, 1996.

ELIZONDO, D. The linear separability problem: Some testing methods. *Neural Networks, IEEE Transactions on*, IEEE, v. 17, n. 2, p. 330–344, 2006.

EVERITT, B. S. *Cluster Analysis*. New York: John Wiley & Sons, 1974.

GARNIER, S. et al. Collective decision-making by a group of cockroach-like robots. In: IEEE. *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*. California, USA, 2005. p. 233–240.

GRASSÉ, P.-P. La reconstruction du nid et les coordinations interindividuelles chezbellicositermes natalensis etcubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Insectes sociaux*, Springer, v. 6, n. 1, p. 41–80, 1959.

GROSS, R.; MAGNENAT, S.; MONDADA, F. Segregation in swarms of mobile robots based on the brazil nut effect. In: *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. St. Louis, MO, USA: IEEE, 2009. p. 4349–4356.

HAMANN, H. *Space-Time Continuous Models of Swarm Robotic Systems*. Austria: Springer, 2010.

HAN, J.; KAMBER, M.; PEI, J. *Data Mining: Concepts and Techniques*. San Francisco, CA, USA: Morgan Kaufmann, 2006.

HEINZELMAN, W.; CHANDRAKASAN, A.; BALAKRISHNAN, H. Energy-efficient communication protocol for wireless microsensor networks. In: *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*. Hawaii: IEEE, 2000. p. 10 pp. vol.2–.

JAIN, A. K.; DUBES, R. C. *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988. ISBN 0-13-022278-X.

KARABOGA, D.; BASTURK, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of global optimization*, Springer, v. 39, n. 3, p. 459–471, 2007.

KAZADI, S.; ABDUL-KHALIQ, A.; GOODMAN, R. On the convergence of puck clustering systems. *Robotics and Autonomous Systems*, Elsevier, v. 38, n. 2, p. 93–117, 2002.

LAURO, A. L. *Agrupamento de Dados Utilizando Algoritmo de Colônia de Formigas*. Dissertação (Mestrado) — Curso de Engenharia Civil, UFRJ/COPPE, Rio de Janeiro, 2008.

LEE, C.; KIM, M.; KAZADI, S. Robot clustering. In: *Systems, Man and Cybernetics, 2005 IEEE International Conference on*. Hawaii: IEEE, 2005. v. 2, p. 1449–1454.

LINDEN, R. Técnicas de agrupamento. *Revista de Sistemas de Informação da FSMA*, v. 1, n. 4, p. 18–36, 2009.

MARJOVI, A.; CHOOBDAR, S.; MARQUES, L. Robotic clusters: Multi-robot systems as computer clusters: A topological map merging demonstration. *Robotics and Autonomous Systems*, Elsevier, v. 60, n. 9, p. 1191–1204, 2012.

MENDONÇA, R. M. de; NEDJAH, N.; MOURELLE, L. de M. *Algoritmos distribuídos para alocação dinâmica de tarefas em enxame de robôs*. Dissertação (Mestrado) — Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2014.

OLFATI-SABER, R.; SHAMMA, J. S. Consensus filters for sensor networks and distributed sensor fusion. In: IEEE. *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*. Sevilla, España, 2005. p. 6698–6703.

PASCUAL, D.; PLA, F.; SÁNCHEZ, S. Algoritmos de agrupamiento. *Método Informáticos Avanzados*, 2007.

ROSAS, R. M. V. et al. Tratamiento del desbalance en problemas con múltiples clases con ecoc. *Computación y Sistemas (CyS)*, v. 17, n. 4, 2013.

ROSATO, A. et al. Why the brazil nuts are on top: Size segregation of particulate matter by shaking. *Physical Review Letters*, APS, v. 58, n. 10, p. 1038, 1987.

RUBENSTEIN, M.; AHLER, C.; NAGPAL, R. Kilobot: A low cost scalable robot system for collective behaviors. In: IEEE. *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. St. Paul, MN, USA, 2012. p. 3293–3298.

SAKHAEE, E.; JAMALIPOUR, A. Stable clustering and communications in pseudolinear highly mobile ad hoc networks. *Vehicular Technology, IEEE Transactions on*, IEEE, v. 57, n. 6, p. 3769–3777, 2008.

VARDY, A.; VOROBYEV, G.; BANZHAF, W. Cache consensus: rapid object sorting by a robotic swarm. *Swarm Intelligence*, Springer US, v. 8, n. 1, p. 61–87, 2014. ISSN 1935-3812. Disponível em: <http://dx.doi.org/10.1007/s11721-014-0091-5>.

XU, R.; WUNSCH, D. C. *Clustering*. Hoboken, NJ, USA: John Wiley & Sons, IEEE Press, 2009.