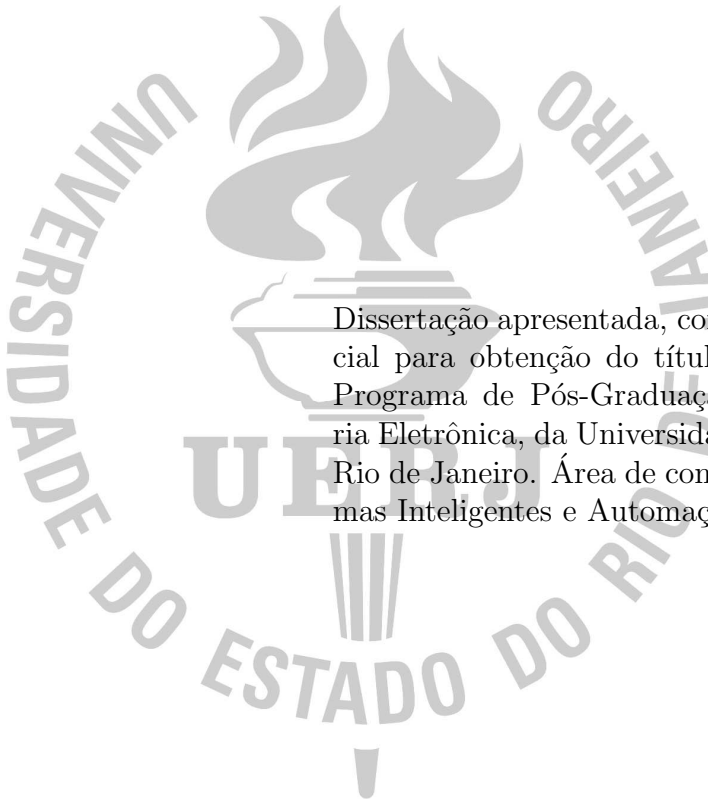
Rafael Mathias de Mendonça

Algoritmos distribuídos para alocação dinâmica de tarefas em enxame de robôs



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
Co-orientadora: Prof^a. Dr^a. Luiza de Macedo Mourelle

Rio de Janeiro
2014

CATALOGAÇÃO NA FONTE
UERJ/REDE SIRIUS/CTC/B

M586 Mendonça, Rafael Mathias
Algoritmos distribuídos para alocação dinâmica de tarefas em enxame de robôs/Rafael Mathias Mendonça. – 2014.
100 f. : il.

Orientadora: Nadia Nedjah.
Co-orientadora: Luiza de Macedo Mourelle.
Dissertação (Mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.
Bibliografia: f. 94 – 100.

1. Redes de computação (Redes embutidas). 2. alocação dinâmica de tarefas. 3. robótica de enxame. 4. computação distribuída. 5. inteligência de enxame. I. Nedjah, Nadia. II. Mourelle, Luiza de Macedo. III. Universidade do Estado do Rio de Janeiro. Faculdade de Engenharia. IV. Título.

CDU 510.5

Assinatura Data

Rafael Mathias de Mendonça

Algoritmos distribuído para alocação dinâmica de tarefas em enxame de robôs

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

Banca Examinadora:

Prof^a. Dr^a. Nadia Nedjah (Orientadora)
Faculdade de Engenharia, UERJ

Prof^a. Dr^a. Luiza de Macedo Mourelle (Co-orientadora)
Faculdade de Engenharia, UERJ

Prof. Dr. Leandro dos Santos Coelho
Faculdade de Engenharia, UFPR e PUCPR

Prof^a. Dr^a. Sheila Regina Murgel Veloso
Faculdade de Engenharia, UERJ

Rio de Janeiro
2014

RESUMO

DE MENDONÇA, Rafael Mathias. *Algoritmos distribuídos para alocação dinâmica de tarefas em enxame de robôs*. 2014. 100f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2014.

A Inteligência de Enxame foi proposta a partir da observação do comportamento social de espécies de insetos e pássaros. A ideia central deste comportamento coletivo é executar uma tarefa complexa decompondo-a em tarefas simples, que são facilmente executadas pelos indivíduos do enxame. A realização coordenada destas tarefas simples, respeitando uma proporção pré-definida de execução, permite a realização da tarefa complexa. O problema de alocação de tarefas surge da necessidade de alocar as tarefas aos indivíduos de modo coordenado, permitindo um bom gerenciamento do enxame. A alocação de tarefas é um processo dinâmico pois precisa ser continuamente ajustado em resposta a alterações no ambiente, na configuração do enxame e/ou no desempenho do mesmo. A robótica de enxame surge deste contexto de cooperação coletiva, ampliada à robôs reais. Nesta abordagem, problemas complexos são resolvidos pela realização de tarefas complexas por enxames de robôs simples, com capacidade de processamento e comunicação limitada. Objetivando obter flexibilidade e confiabilidade, a alocação deve emergir como resultado de um processo distribuído. Com a descentralização do problema e o aumento do número de robôs no enxame, o processo de alocação adquire uma elevada complexidade. Desta forma, o problema de alocação de tarefas pode ser caracterizado como um processo de otimização que aloca as tarefas aos robôs, de modo que a proporção desejada seja atendida no momento em que o processo de otimização encontre a solução desejada. Nesta dissertação, são propostos dois algoritmos que seguem abordagens distintas ao problema de alocação dinâmica de tarefas, sendo uma local e a outra global. O algoritmo para alocação dinâmica de tarefas com abordagem local (ADTL) atualiza a alocação de tarefa de cada robô a partir de uma avaliação determinística do conhecimento atual que este possui sobre as tarefas alocadas aos demais robôs do enxame. O algoritmo para alocação dinâmica de tarefas com abordagem global (ADTG) atualiza a alocação de tarefas do enxame com base no algoritmo de otimização PSO. No ADTG, cada robô possui uma possível solução para a alocação do enxame que é continuamente atualizada através da troca de informação entre os robôs. As alocações são avaliadas quanto a sua aptidão em atender à proporção objetivo. Quando é identificada a alocação de maior aptidão no enxame, todos os robôs do enxame são alocados para as tarefas definidas por esta alocação. Os algoritmos propostos foram implementados em enxames com diferentes arranjos de robôs reais demonstrando sua eficiência e eficácia, atestados pelos resultados obtidos.

Palavras-chave: alocação dinâmica de tarefas, robótica de enxame, computação distribuída, inteligência de enxame.

ABSTRACT

Swarm Intelligence has been proposed based on the observation of social behavior of insect species and birds. The main idea of this collective behavior is to perform a complex task decomposing it into many simple tasks, that can be easily performed by individuals of the swarm. Coordinated realization of these simple tasks while adhering to a pre-defined distribution of execution, allows for the achievement of the original complex task. The problem of task allocation arises from the need of assigning tasks to individuals in a coordinated fashion, allowing a good management of the swarm. Task allocation is a dynamic process because it requires a continuous adjustment in response to changes in the environment, the swarm configuration and/or the performance of the swarm. Swarm robotics emerges from this context of collective cooperation applied to swarms of real robots. In this approach, complex problems are solved by performing complex tasks using swarms of simple robots, with a limited processing and communication capabilities. Aiming at achieving flexibility and reliability, the allocation should emerge as a result of a distributed process. With the decentralization of the problem and the increasing number of robots in the swarm, the allocation process acquires a high complexity. Thus, the problem of task allocation can be characterized as an optimization process that assigns tasks to robots, so that the desired proportion is met at the end of the optimization process, find the desired solution. In this dissertation, we propose two algorithms that follow different to the problem of dynamic task allocation approaches: one is local and the other global. The algorithm for dynamic allocation of tasks with a local approach (ADTL) updates the task assignment of each robot based on a deterministic assessment of the current knowledge it has so far about the tasks allocated to the other robots of the swarm. The algorithm for dynamic task allocation with a global approach (ADTG) updates the allocation of tasks based on a swarm optimization process, inspired by PSO. In ADTG, each robot has a possible solution to the swarm allocation, which is continuously updated through the exchange of information between the robots. The allocations are evaluated for their fitness in meeting the goal proportion. When the allocation of highest fitness in the swarm is identified, all robots of the swarm are allocated to the tasks defined by this allocation. The proposed algorithms were implemented on swarms of different arrangements of real robots demonstrating their efficacy, robustness and efficiency, certified by obtained the results.

Keywords: Dynamic task allocation, swarm robotics, distributed computing, swarm intelligence.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE SIGLAS

δ	Diferença $\mathbb{C} - \mathbb{C}_i$ entre o contador $\mathbb{C}$ e o contador $\mathbb{C}_i$
$\mathbb{A}$	Alocação de tarefas para o enxame
$\mathbb{A}_i$	Alocação de tarefas atual do enxame, para o robô i
$\mathbb{C}$	Conjunto dos τ contadores do número de robôs alocados a cada tarefa
$\mathbb{C}_i$	Conjunto dos τ contadores do número de robôs alocados a cada tarefa, para o robô i
$\mathbb{I}$	Conjunto dos identificadores dos robôs do enxame
$\mathbb{P}$	Proporção Objetivo
$\mathbb{P}_i$	Proporção atual do enxame, para o robô i
$\mathbb{T}$	Conjunto de identificadores das tarefas alocadas aos robôs
ρ	Número de robôs no enxame
τ	Número de tarefas para alocar
id	Identificador do robô
ADADiT	Algoritmo Determinístico para Alocação Dinâmica de Tarefas
ADT	Alocação Dinâmica de Tarefas

SUMÁRIO

INTRODUÇÃO

Sistemas multi-robôs (SMR), podem ser descritos como um conjunto de múltiplos robôs, executando tarefas de forma coordenada, em busca de um objetivo comum. Nas últimas décadas tem-se observado um rápido crescimento das aplicações de sistemas multi-robôs. Atualmente os pesquisadores em robótica vêm dirigindo seus esforços para a construção de robôs móveis e introduzindo novas capacidades nos robôs afim de torná-los mais independentes dos humanos na realização de suas atividades.

Diversas aplicações à esta proposta podem ser vislumbradas. Por exemplo, em situações que representem um risco ou inviabilizem a presença humana para a execução de uma tarefa, o enxame de robôs seria capaz de se auto-organizar para formar grupos, onde cada grupo executaria uma tarefa específica com o objetivo, de juntos realizarem uma ação significativa. Com a adição do elemento dinâmico de alocação de tarefas, o enxame ganharia uma maior flexibilidade e confiabilidade.

Utiliza-se MRS intensamente na solução de problemas complexos devido a suas características apreciáveis à resolução deste tipo de problema (BAYINDIR; SAHIN, 2007). Sua abordagem tem como inspiração sistemas cooperativos, tais como apresentado em espécies sociais, que demonstram três características desejadas para os sistemas multi-robô: robustez, flexibilidade e escalabilidade.

A *robustez* pode ser definida como o grau em que um sistema pode continuar a funcionar na presença de falhas parciais ou outras condições anormais. Para grupos de robôs cooperativos esta característica permite ao enxame superar dificuldades em ambientes dinâmicos e imprevisíveis.

Define-se *flexibilidade* como a capacidade do sistema em se adaptar a uma nova situação ou problema mantendo-se a mesma base de mecanismos de auto-organização. Em alguns momentos, flexibilidade e robustez possuem definições em parte conflitantes. A flexibilidade de um sistema refere-se a um nível de abstração mais elevado quando

comparado a robustez. Desta forma, dizemos que um sistema é flexível quando o mesmo possui aplicabilidade à resolução de mais de um problema sem que as características básicas do sistema sejam alteradas. Os sistemas biológicos são um bom exemplo de sistema flexível uma vez que podem facilmente adaptar os seus comportamentos em resposta a diferentes problemas.

A *escalabilidade* é definida como a capacidade do sistema em permitir a sua expansão, suportando a inclusão de um número maior ou menor de indivíduos, sem que ocorra uma degradação significativa no seu desempenho. Esta é uma característica de muitos algoritmos evolucionários bio-inspirados que baseiam o processo de otimização na existência de indivíduos, onde cada indivíduo representa uma possível solução para o problema. Para a aplicação em grupos de robôs a escalabilidade de um enxame esta diretamente ligada a capacidade exploratória do sistema uma vez que um maior número de indivíduos possibilita uma maior gama de possíveis soluções para o problema. Contudo a escolha de um sistema com alta escalabilidade pode ocasionar uma degradação no desempenho do sistema, principalmente na comunicação. Desta forma muitas pesquisas tem sido direcionadas à ponderação entre a escalabilidade e a comunicação buscando níveis aceitáveis de desempenho para o sistema.

As pesquisas desenvolvidas atualmente em sistemas multi-robôs apresentam-se organizadas em nove diferentes campos de pesquisa, classificados em (PARKER, 2003) e (MOHAN; PONNAMBALAM, 2009). Assim teremos como campos de pesquisa: Inspiração Biológica, Comunicação, Controle, Mapeamento e Localização, Transporte e Manipulação, Robótica Reconfigurável, Coordenação de Movimentos, Aprendizagem e Alocação de Tarefas.

No campo de pesquisa da *Inspiração Biológica*, as espécies sociais de insetos fornecem um dos mais conhecidos exemplos de comportamento biológico auto-organizado. Os trabalhos realizados neste campo com grupos cooperativos de robôs tem demonstrado resultados significativos ao resolver problemas complexos inspirando-se nas habilidades destas espécies (DROGOUL; FERBER, 1993) (MATARIC, 1993). Os principais tópicos de relevância incluem a observação da habilidade de grupos de animais em aprender novos conhecimentos e a interconectividade física demonstrada por espécies sociais de insetos em realizar a navegação do grupo através de terrenos de difícil locomoção.

A questão da *Comunicação* em grupos de robôs foi inicialmente estudada a partir de pesquisas de algoritmos distribuídos. De acordo com os mecanismos de comunicação empregados pelo enxame a comunicação pode ser considerada implícita ou explícita. Na comunicação implícita a comunicação entre os robôs pode se dar através de processos indiretos de comunicação (sinais luminosos, sinais sonoros), como mostrado em (MENDONÇA; NEDJAH; MOURELLE, 2012). Na comunicação explícita é exigida uma comunicação dada por uma ação específica e direcionada para a transmissão de informações entre os robôs (radio frequência, bluetooth). Os mais recentes trabalhos neste campo estão ligados a representação da linguagem e na fundamentação destas representações no mundo real (JUNG; ZELINSKY, 2000).

O *Controle* em grupos multi-robôs pode apresentar-se de forma distribuída ou centralizada. Como bem definido em (IOCCHI; NARDI; SALERNO, 2001) e (JR; THOMAS, 2007), o controle centralizado necessita de um agente central encarregado de centralizar as informações e organizar o trabalho dos demais agentes. Para a definição de um controle como centralizado é necessário que o processo de decisão esteja atrelado a apenas um agente central, diferenciado dos demais que agem de acordo com as orientações deste. No caso do controle distribuído não existe a presença de um agente central estando o processo de decisão completamente autônomo entre os agentes.

Na aplicação à robótica de enxame, a escolha da forma de controle esta intimamente ligada com as necessidades de cada problema. Na abordagem centralizada resultados ótimos podem ser produzidos uma vez que o robô central tem acesso a todas as informações no seu processo de decisão. Contudo o processo de decisão centralizado é fortemente baseado em comunicação tornando-o mais vulnerável a falhas na comunicação e falhas de funcionamento do robô central. Além do fato que a resposta do sistema em relação a mudanças no ambiente costuma ser lento uma vez que todas as informações relevantes devem ser encaminhadas para o robô central antes que qualquer ação possa ser realizada.

Em contrapartida, na abordagem distribuída ao fornecer autonomia no processo de decisão aos robôs do enxame, obtém-se respostas mais rápidas do sistema na ocorrência de mudanças no ambiente e torna o sistema menos vulnerável as falhas, agregando robustez ao mesmo. Outro ponto forte na abordagem distribuída é o seu inerente paralelismo

que torna o sistema mais escalável permitindo a realização de tarefas mais rapidamente. Entretanto a distribuição do processo de decisão algumas vezes resulta em soluções subótimas uma vez que as decisões baseiam-se em informações locais e em alguns momentos incompletas. Outro problema encontra-se na coordenação de atividades entre os robôs para a realização de uma tarefa. Nestes casos, um processo de sincronismo para a troca de informações entre os robôs, normalmente, é necessário tornando o sistema como um todo mais complexo.

Muitas pesquisa tem sido realizadas no campo de *Mapeamento e Localização* nas últimas duas décadas. O mapeamento é uma representação dos ambientes físicos através de dados sensoriais em modelos espaciais fornecidos por robôs móveis (THRUN, 2002). Enquanto que a localização é definida como o processo de encontrar a localização do robô nos modelos espaciais gerados. Atualmente este campo de pesquisa está concentrado em duas áreas: SLAM, abreviação de Simultaneous Localization and Mapping, e CML, abreviação de Concurrent Mapping and Localization. As duas áreas referem-se ao problema de aquisição de um mapa em um ambiente desconhecido com um robô móvel, enquanto é realizada a localização simultânea do robô em relação a este mapa (THRUN, 2002).

Em *Transporte e Manipulação* utiliza-se um enxame de robôs para desempenhar ações cooperativas de manipulação de objetos, tais como transportar, empurrar ou puxar (RUS; DONALD; JENNINGS, 1995) (STILWELL; BAY, 1993). Esta área de pesquisa possui um amplo cenário de aplicações práticas que a torna particularmente interessante para estudo. Praticamente todos os trabalhos desenvolvidos para esta área envolvem o transporte de objetos por robôs móveis através de superfícies planas. Um desafio em aberto para esta área seria o transporte cooperativo sob terrenos irregulares em campo aberto.

Na *Robótica Reconfigurável* os robôs são tratados como máquinas modulares e autônomas com uma morfologia variável tendo a capacidade de auto-reconfiguração em resposta a mudanças no ambiente. Além das características usuais normalmente encontradas em robôs de morfologia fixa, os robôs auto-reconfiguráveis são capazes de alterar deliberadamente a estrutura de suas formações com o objetivo de atender as novas circunstâncias do problema, realizar novas tarefas ou mesmo recuperar-se de danos sofridos (RYLAND; CHENG, 2010). Pesquisas nesta área ainda são muito recentes sendo considerado um campo rico para contínuos avanços em sistemas multi-robôs.

A *Coordenação de Movimentos* é um tópico de estudo com bastante aceitação do meio acadêmico onde são encontrados diversos trabalhos. A abordagem mais habitual é definida como *path-planning*, onde é descrito um ambiente destacando duas posições, uma descrita como inicio e outra como final. Estando o robô inicialmente posicionado na posição de início o mesmo deve especificar e realizar uma rota de deslocamento entre o ponto de início e o de final desviando dos obstáculos encontrados e respeitando certos critérios de otimização durante o seu deslocamento (GE; CUI, 2000). Outra abordagem de grande aceitação trata-se da *formação* que define a organização de um grupo de robôs de maneira coordenada para iniciar e manter uma formação física específica como mostrado em (SAHIN et al., 2002).

Descrevemos a *Aprendizagem* em MRS, como a ação de adquirir conhecimento através de informações obtidas no ambiente ou na observação de suas próprias ações com o objetivo de modificar suas ações futuras buscando a melhoria de algum aspecto. A aprendizagem é classificada como supervisionada ou não-supervisionada. Na aprendizagem supervisionada o robô adquire conhecimentos para a resolução de algum problema através de um agente externo, dito supervisor, que fornece as informações corretivas para o robô. Em oposição, na aprendizagem não-supervisionada o robô adquire conhecimento sem nenhuma, ou mesmo com pouca, informação externa de caráter corretiva. A utilização da aprendizagem não-supervisionada permite que os robôs se adaptem a situações em que se desconhece a tarefa ou o ambiente de uma forma mais rápida e eficiente (PUGH; MARTINOLI, 2006).

Por último, e não menos importante, temos a área de pesquisa que trata da *Alocação de Tarefas*. A alocação de tarefas é parte integrante na resolução de um problema complexo, onde inicialmente este problema é decomposto em um número de tarefas de simples execução por robôs de capacidade limitada. A tarefa é definida como um sub-objetivo necessário para a realização do objetivo global. As execuções destas tarefas são definidas respeitando-se uma proporção de execução das tarefas entre os robôs do enxame, definida na etapa de decomposição. A seguir, as tarefas são devidamente alocadas, respeitando-se a proporção estabelecida, para serem executadas pelos robôs do enxame. Dessa forma, através de uma ação cooperativa entre os robôs do enxame é alcançada uma solução para o problema complexo.

Compete a etapa de alocação de tarefas atribuir as tarefas entre os membros do grupo de forma produtiva e eficiente. A alocação deve garantir que não só o objetivo global seja alcançado, mas também que as tarefas sejam bem distribuídas entre os robôs. Uma abordagem eficaz da alocação considera os recursos disponíveis, os aspectos a se otimizar (tempo, convergência) e as capacidades individuais dos robôs de grupos heterogêneos em executar as tarefas (BAGHAEI; AGAH, 2002).

Diversos algoritmos de alocação de tarefas tem sido propostos sob influência de diferentes metodologias através das abordagens comportamentais, baseadas nas leis de mercado e bio-inspiradas. Nestas abordagens é notável a influência de processos de controle baseados em sistemas distribuídos tendo em vista a sua flexibilidade e robustez ao ser empregado na solução de problemas desta natureza.

É possível encontrar muitas arquiteturas de coordenação multi-robô, mas relativamente poucos modelos formais. Em (CAO et al., 1995) e (BALCH; PARKER, 2002) encontramos um tratamento completo da arquitetura de sistemas multi-robôs. Cada um destes oferece uma taxonomia que classifica a maior parte dos sistemas multi-robô existentes ao longo de vários eixos, como a organização do grupo (centralizado ou distribuído), a topologia de comunicação, a composição do grupo (homogêneo ou heterogêneo), o tipo de cooperação (enxame e intencional).

No tipo de cooperação por enxame tratamos da cooperação de um enxame homogêneo composto por um grande número de robôs com capacidade limitada. No entanto, quando muitos desses robôs são reunidos, um comportamento globalmente interessante pode surgir como resultado das interações locais entre os robôs. Essa abordagem normalmente dependem de resultados de convergência matemáticos que indicam o resultado desejado ao longo de um período suficientemente longo de tempo envolvendo várias repetições de uma mesma atividade. O segundo tipo trata da cooperação intencional entre um número limitado de robôs tipicamente heterogêneos realizando várias tarefas distintas. Neste, os robôs muitas vezes precisam lidar com algum tipo de restrição para atingir a convergência e requerem a utilização de um tipo mais proposital de cooperação durante a execução das tarefas. A dificuldade está em determinar qual robô deve executar cada tarefa, de modo a maximizar a eficiência do grupo e assegurar a coordenação adequada entre os membros.

Ao invés de caracterizar arquiteturas, em (GERKEY; MATARIĆ, 2004) é abordada uma classificação mais geral do problema de alocação apresentando uma taxonomia particular:

- Single-task robots (ST) e multi-task robots (MT): ST define que cada robô é capaz de executar apenas uma tarefa por vez, enquanto MT define que um ou mais robôs são capazes de executar mais de uma tarefa simultaneamente.

- Single-robot tasks (SR) e multi-robot tasks (MR): SR define que cada tarefa requer apenas um robô para ser realizada, enquanto MR define que ao menos uma das tarefas requer mais de um robô para ser realizada.

- Instantaneous assignment (IA) e Time-extended assignment (TA): IA define que as informações disponíveis são limitadas e permitem apenas alocações instantâneas, sem que haja um planejamento para futuras alocações, enquanto TA define que as informações disponíveis são mais completas, tais como o conjunto de todas as tarefas que precisam ser atribuídas, permitindo um planejamento futuro.

Sistemas distribuídos são sistemas autônomos, normalmente interconectados por uma rede de comunicação, que aparecem para o usuário como um sistema único com a finalidade de realizar uma ação. Nesta abordagem, aparecem os algoritmos distribuídos (LYNCH, 1996) que são executados de modo concorrente em cada individuo. Quando os algoritmos precisam cooperar entre si para realizar uma ação, estes coordenam suas ações através da troca de informações. A troca de informação acontece em momentos definidos no algoritmo onde são atualizadas informações necessárias para a continuidade da execução do programa em cada indivíduo. A concorrência de execuções, permite ao sistema a possibilidade de adaptar-se a ocorrência de falhas de execução em algum dos algoritmos distribuídos. Esta característica possibilita a realização da ação mesmo em condições adversas, como falhas de comunicação e mudanças no ambiente.

Da mesma forma, o problema de alocação de tarefas em grupos multi-robôs, é um processo dinâmico pois precisa ser continuamente ajustado em resposta às alterações no ambiente e/ou no desempenho do enxame (GERKEY; MATARIĆ, 2004). Uma solução imediata para resolver este problema baseia-se na abordagem centralizada. No entanto, uma alocação distribuída representa uma melhor aproximação do comportamento dos

enxames de espécies sociais, onde não existe um mecanismo de controle centralizado. Portanto, a alocação de tarefas em enxames de robôs deve surgir como resultado de um processo distribuído. Esta descentralização aumenta a complexidade do problema, pois o robô não tem uma visão completa do ambiente. Cada robô deverá tomar decisões de controle locais sem o conhecimento completo do que outros robôs fizeram no passado, estão fazendo agora ou irão fazer no futuro.

A alocação de tarefas é um processo dinâmico pois precisa ser continuamente ajustada em resposta às alterações no ambiente e/ou no desempenho do enxame. Uma solução imediata para resolver este problema baseia-se na abordagem centralizada.

No entanto, uma alocação distribuída representa uma melhor aproximação do comportamento dos enxames de espécies sociais, onde não existe um mecanismo de controle centralizado. Portanto, a alocação de tarefas em enxames de robôs deve surgir como resultado de um processo distribuído. Esta descentralização aumenta a complexidade do problema, pois o robô não tem uma visão completa do ambiente. Cada robô deverá tomar decisões de controle locais sem o conhecimento completo do que outros robôs fizeram no passado, estão fazendo agora ou irão fazer no futuro.

Para superar tal dificuldade, são propostos dois algoritmos para alocação dinâmica de tarefas em um enxame de robôs. Os algoritmos são descentralizados e bastante simples, permitindo uma execução eficiente em robôs dotados de recursos de armazenamento e processamento limitados.

No Capítulo 1 é apresentado o problema da alocação dinâmica de tarefas através de uma definição formal do mesmo. É realizada também uma análise a respeito da complexidade do enxame para diferentes arranjos. A seguir são apresentadas as estratégias de alocação de tarefas de acordo com o tipo de controle, o método de decisão e o método de execução.

O Capítulo 2 apresenta um estudo sobre os principais trabalhos realizados no estudo de algoritmos para a alocação de tarefas em enxame de robôs. Diferentes metodologias são abordadas buscando realizar a alocação de tarefas de uma forma mais eficiente.

No Capítulo 3 é proposto um algoritmo determinístico de alocação dinâmica de tarefas com uma abordagem local a respeito da alocação do enxame. São abordadas as etapas do algoritmo, bem como sua estrutura de execução. Nesta proposta, cada robô

atualiza a sua alocação de tarefa a partir do conhecimento da alocação dos demais robôs, através da troca de informações entre eles.

No Capítulo 4 é proposto um algoritmo estocástico de alocação dinâmica de tarefas com uma abordagem global a respeito da alocação de tarefas do enxame. São abordadas as etapas do algoritmo, bem como sua estrutura de execução. O algoritmo *PSO* é utilizado como inspiração para o processo de otimização. Nesta proposta, o problema de alocação é tratado como um problema de otimização, de modo que ao término do processo de otimização é definida uma alocação de tarefas global para o enxame.

O Capítulo 5 apresenta a implementação dos algoritmos propostos em enxames de robôs reais. É descrito os detalhes do *hardware* do robô utilizado bem com seus aspectos de comunicação. O processo de comunicação consiste da troca de mensagens entre os robôs do enxame utilizando a comunicação por rádio frequência. São abordados os detalhes deste processo de comunicação, bem como a estrutura das mensagens geradas.

O Capítulo 6 exibe os resultados obtidos pelos ensaios realizados com enxames de diferentes arranjos. Os resultados obtidos são avaliados e comparados a partir de aspectos de convergência e comunicação com o objetivo de avaliar a eficácia e eficiência de cada algoritmo proposto. No aspecto de convergência, é analisado o tempo necessário para que o enxame apresente uma alocação que atenda a proporção-objetivo. O número de mensagens trocadas durante os processos de comunicação é utilizado como forma de analisar os resultados quanto ao aspecto de comunicação.

Por fim, no Capítulo 7 são apresentadas as considerações finais desta dissertação a partir de uma análise dos resultados obtidos durante os ensaios realizados. Após são apresentadas propostas de melhoria para os algoritmos propostos e sugestões para trabalhos futuros.

Capítulo 1

ALOCAÇÃO DINÂMICA DE TAREFAS

ESTE capítulo introduz o problema da alocação dinâmica de tarefas através de uma definição formal do problema e uma análise da sua complexidade. A alocação de tarefas apresenta-se como uma funcionalidade operacional necessária para o gerenciamento de enxames de robôs.

A Seção 1.1 define o problema da alocação dinâmica de tarefas. A Seção 1.2 apresenta uma análise da complexidade do processo de alocação de tarefas. Na Seção 1.3 são apresentadas as estratégias usadas para alocação de tarefas. Na Seção 1.4 são realizadas as considerações finais para o capítulo.

1.1 Definição do Problema

Compete ao processo de Alocação Dinâmica de Tarefas (ADT) a ação de alocar as tarefas aos robôs de uma forma eficiente. Qualquer abordagem de alocação deve considerar os recursos disponíveis, tais como meios de comunicação, as características a se otimizar, tais como eficácia e eficiência e as capacidades individuais dos robôs de grupos heterogêneos em realizar as tarefas (BAGHAEI; AGAH, 2002).

Geralmente, um enxame de robôs tem como objetivo realizar uma tarefa complexa que é dividida em várias outras tarefas menos complexas. Este processo é chamado de *decomposição*. O processo de decomposição identifica o conjunto de tarefas válidas, e a proporção de alocação ideal para que o objetivo original seja alcançado. Dessa forma, na decomposição são definidos os parâmetros necessários ao processo de alocação de tarefas.

Com o objetivo de fornecer uma definição formal do problema de alocação dinâmica de tarefas, seja $\mathbb{T} = \{t_1, t_2, \ldots, t_\tau\}$ o conjunto de identificadores de tarefas a serem alocadas aos robôs do enxame, sendo este composto pelas τ tarefas válidas a serem realizadas pelo enxame. Considerando um enxame composto por ρ robôs, o processo de alocação de tarefas permite a alocação das τ tarefas aos ρ robôs do enxame, respeitando-se a proporção desejada $\mathbb{P}$. A proporção desejada $\mathbb{P} = \{p_1, p_2, \ldots, p_\tau\}$, é definida por um conjunto de inteiros positivos $p_i \in \mathbb{N}^*$, tais que:

$$\sum_{i=1}^{\tau} p_i = 1, \tag{1}$$

sendo, $c_i = p_i \times \rho$ robôs devem ser alocados à tarefa t_i.

O número de robôs alocados a cada uma das tarefas é representado por um conjunto de contadores $\mathbb{C} = \{c_1, c_2, \ldots, c_\tau\}$, tal que:

$$\sum_{i=1}^{\tau} c_i = \rho. \tag{2}$$

O conjunto $\mathbb{C}$ define a quantidade de robôs alocados a cada umas das τ tarefas, de forma que o número de robôs alocados a tarefa $t_i \in \mathbb{T}$ corresponde ao contador $c_i \in \mathbb{C}$.

No enxame, cada robô possui uma identificação única definida como id, que o diferencia dos demais. No restante desta dissertação o conjunto de identificadores dos robôs do enxame será representado por $\mathbb{I} = \{id_1, id_2, \ldots, id_\rho\}$.

A alocação do enxame é representada por $\mathbb{A} = \{a_1, a_2, \ldots, a_\rho\}$, onde a_i identifica a tarefa alocada ao robô id_i. A partir de uma alocação $\mathbb{A}$ do enxame é possível deduzir os elementos do conjunto $\mathbb{C}_\mathbb{A}$ de contadores, conforme a Equação 3:

$$c_i = \mathbb{C}_\mathbb{A}[t_i] = \sum_{r=1}^{\rho} \phi(a_r, t_i), \tag{3}$$

onde a função ϕ é definida conforme a Equação 4:

$$\phi(a, t) = \begin{Bmatrix} 1 \ se \ a = t \\ 0 \ senão \end{Bmatrix} \tag{4}$$

de forma que ao término da contabilização o conjunto $\mathbb{C}_\mathbb{A}$ representará a distribuição de robôs alocados em cada tarefa definido pela alocação $\mathbb{A}$.

A resolução do problema de alocação dinâmica de tarefa consiste em encontrar a alocação $\mathbb{A}^* = \{a_1^*, a_2^*, \ldots, a_\rho^*\}$ tal que:

$$\forall\ t_i \in \mathbb{T}\ e\ \forall\ p_i \in \mathbb{P},\ \mathbb{C}_{\mathbb{A}^*}[t_i] = p_i \times \rho \tag{5}$$

1.2 Análise de Complexidade

A complexidade do processo de alocação de tarefas é definido por um problema de decisão em que se deseja encontrar a alocação $\mathbb{A}^*$ que resolva o problema de ADT pertencente a um conjunto de alocações factíveis $\mathbb{Q}$. O número de alocações factíveis $\mathbb{Q}$ depende do número de robôs ρ do enxame e de tarefas τ a serem alocadas de acordo com as características do enxame. O problema de ADT apresenta uma complexidade computacional alta sendo classificado como *NP_completo*.

A complexidade do processo de alocação de tarefas é diretamente influenciada pela composição do enxame. De acordo com a sua composição, um enxame pode ser *homogêneo* ou *heterogêneo*. O enxame homogêneo é constituído por um único grupo de ρ robôs que possuem as mesmas características e portanto são capazes de realizar um mesmo conjunto de tarefas $\mathbb{T}$. O número de alocações factíveis $\#\mathbb{Q}$ para um enxame homogêneo é dado pela Equação 6:

$$\#\mathbb{Q} = \tau^\rho. \tag{6}$$

Considerando um enxame heterogêneo constituído por ρ robôs, diferenciados em n grupos, sendo cada grupo i composto por ρ_i robôs, tal que:

$$\sum_{i=1}^{n} \rho_i = \rho. \tag{7}$$

Os grupos são formados a partir de uma avaliação do conjunto $\mathbb{T}_i$ de tarefas que os robôs deste grupo são capazes de realizar. Os robôs de grupos diferentes possuem características distintas e são capazes de realizar diferentes conjuntos de tarefas $\mathbb{T}_i$. De maneira complementar, para cada grupo i, os ρ_i robôs deste grupo possuem as mesmas características e são capazes de realizar o mesmo conjunto de tarefas $\mathbb{T}_i$. O número de alocações factíveis $\#\mathbb{Q}$, neste caso, é dado pela Equação 8:

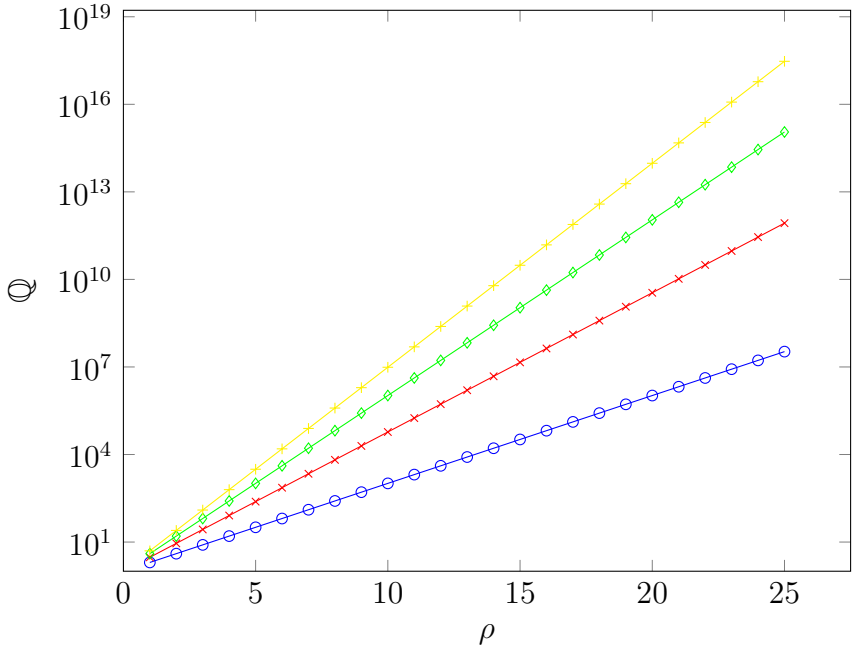


Figura 1: Número de alocações factíveis para um enxame de composição homogênea

$$\#\mathbb{Q} = \prod_{i=1}^{n} \tau_i^{\rho_i} = \tau_1^{\rho_1} \times \tau_2^{\rho_2} \times \cdots \times \tau_n^{\rho_n} \tag{8}$$

onde $\#\mathbb{Q}$ é definido pelo produtório dos números de alocações factíveis entre os n grupos distintos de robôs.

Na Figura 1 é mostrada a relação entre o número de alocações factíveis $\mathbb{Q}$ formadas pelos enxames de composição homogênea. A Figura 2 apresenta um enxame de composição heterogênea formado por dois grupos distintos.

Nas duas situações, o número de alocações factíveis aumenta de acordo com o número de tarefas impostas ao problema, de modo que um maior número de tarefas representa um maior número de combinações de alocações possíveis. Contudo, é perceptível que o número de alocações possíveis é maior para o enxame de composição homogênea quando comparado ao de composição heterogênea. Dessa forma, do ponto de vista de um problema de otimização um enxame homogêneo representa um problema mais complexo que o enxame heterogêneo, devido ao mesmo possuir um espaço de busca maior.

Na Figura 3 é mostrado um exemplo ilustrativo, para enxames com cinco diferentes constituições, sendo um homogêneo e os demais heterogêneos. Neste exemplo, o problema é constituído de um conjunto de tarefas $\mathbb{T} = \{1, 2, 3, 4, 5\}$ a serem alocadas entre os

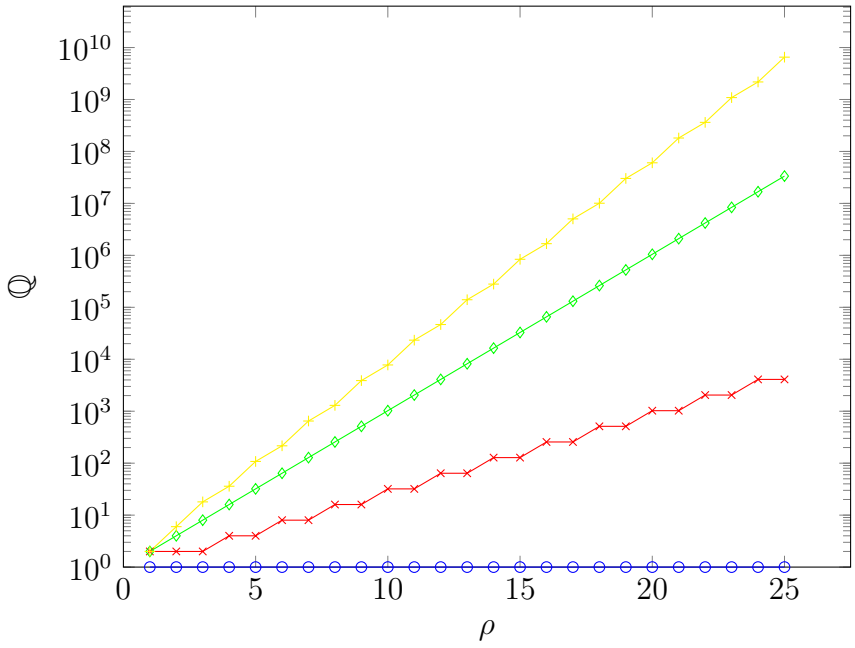


Figura 2: Número de alocações factíveis para um enxame de composição heterogênea

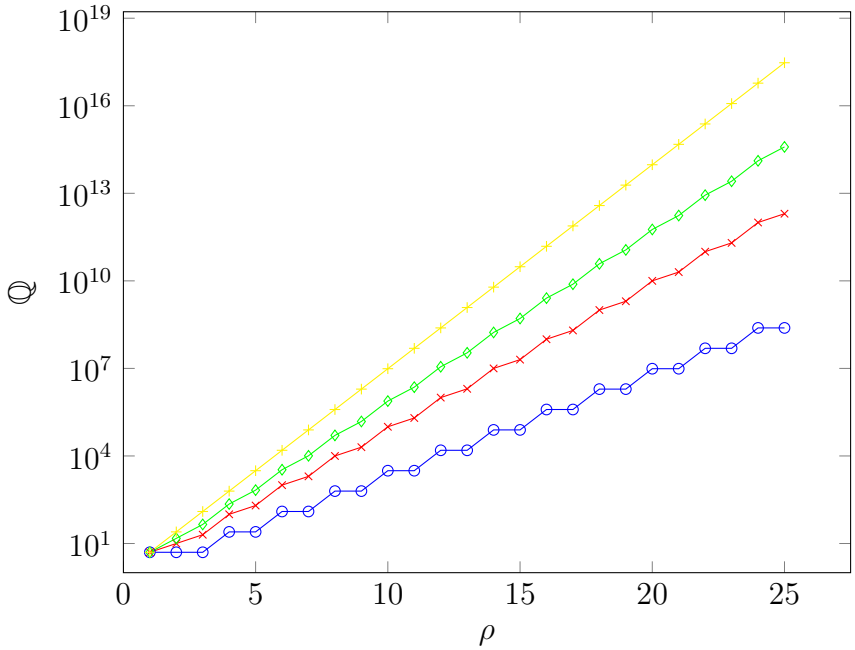


Figura 3: Número de alocações factíveis para enxames de diferentes características

robôs do enxame. Os enxames heterogêneos são formados por dois grupos, sendo cada grupo capaz de realizar o conjunto de tarefas $\mathbb{T}_1$ e $\mathbb{T}_2$, respectivamente. A quantidade de robôs do enxame é dividida igualmente entre cada grupo nos enxames heterogêneos, sendo considerada uma diferença máxima unitária para os casos de enxames com o número total de robôs impar.

O enxame homogêneo é formado por um único grupo capaz de realizar todas as tarefas definidas pelo conjunto $\mathbb{T}$. Dessa forma, é possível observar que o número de alocações factíveis $\#\mathbb{Q}$ é influenciada pela constituição dos grupos em enxames heterogêneos. Independente da constituição dos grupos nos enxames heterogêneos, o enxame homogêneo possui um maior número de alocações factíveis quando comparado aos enxames heterogêneos.

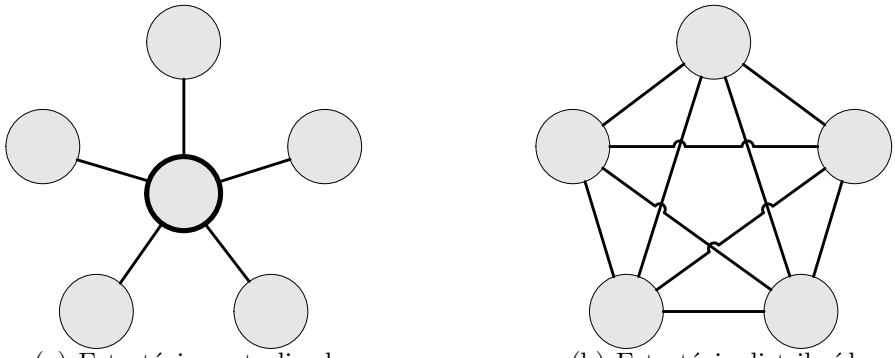
1.3 Estratégias de Alocação de Tarefas

As estratégias de alocação definem a metodologia empregado durante o processo de ADT. As mesmas são classificadas por categorias, de acordo com o tipo de controle, o método de decisão e o método de execução usados. Quanto ao tipo de controle, a alocação pode apresentar-se através de uma estratégia centralizada ou distribuída, conforme mostra a Seção 1.3.1. Quanto o método de decisão, a alocação pode apresentar-se através da estratégia de decisão *determinística* ou *estocástica*, conforme descreve a Seção 1.3.2. Quanto ao método de execução, a locação pode ser executada através de uma estratégia *sequencial* ou *paralela*, conforme mostra a Seção 1.3.3.

1.3.1 Estratégia Centralizada *vs* Estratégia Distribuída

A estratégia de controle em grupos multi-robôs pode apresentar-se de forma centralizada ou distribuída. Como bem definido em (IOCCHI; NARDI; SALERNO, 2001) e (JR; THOMAS, 2007), a estratégia de controle centralizada, ilustrada na Figura 4(a), necessita de um agente central encarregado de reunir as informações e organizar o trabalho, inclusive de alocação de tarefas, aos demais agentes. Para a definição de uma estratégia como centralizada é necessário que o processo de decisão de ADT esteja atrelado a apenas um agente central, diferenciado dos demais que agem de acordo com as orientações deste.

No caso da estratégia de controle distribuída, ilustrada na Figura 4(b), não existe a figura de um agente central. Dessa forma, o processo de decisão de ADT é completamente interno aos agentes.



(a) Estratégia centralizada (b) Estratégia distribuída

Figura 4: Ilustração das estratégias de controle

Na robótica de enxame, a escolha da estratégia de controle está intimamente ligada as necessidades de cada problema. Na estratégia centralizada, resultados ótimos podem ser produzidos uma vez que o robô central tem acesso a todas as informações no seu processo de decisão de alocação. Contudo, a estratégia centralizada é fortemente baseada em comunicação tornando-a mais vulnerável a falhas na comunicação e falhas de funcionamento do robô central. Além do fato que a resposta do sistema em relação às mudanças no ambiente costuma ser lenta uma vez que todas as informações relevantes devem ser encaminhadas para o robô central antes que qualquer ação de alocação de tarefas possa ser iniciada.

Em contrapartida, na estratégia distribuída ao fornecer autonomia no processo de decisão de alocação aos robôs do enxame, obtém-se respostas mais rápidas do sistema na ocorrência de mudanças no ambiente e torna o sistema menos vulnerável as falhas, agregando robustez ao mesmo. Outro ponto forte na estratégia distribuída é o seu inerente paralelismo que torna o sistema mais escalável permitindo que o processo de alocação de tarefas seja realizado mais rapidamente quando comparado a estratégia centralizada. Entretanto, a distribuição do processo de decisão de alocação algumas vezes resulta em soluções sub-ótimas uma vez que as decisões baseiam-se em informações locais e em alguns momentos incompletas. Outro problema encontra-se na coordenação de atividades entre os robôs para a realização de uma tarefa. Nestes casos, um processo de sincronismo para

a troca de informações entre os robôs, geralmente, é necessário tornando o sistema como um todo mais complexo.

1.3.2 Abordagem Determinística *vs* Abordagem Estocástica

Na abordagem determinística, o processo de decisão para a alocação de tarefas é baseado em eventos não probabilísticos. Os processos determinísticos apresentam um comportamento previsível, sendo o resultado de cada etapa que os compõem definido de maneira única. Garantido que os parâmetros da operação não se alteram no tempo, pode-se afirmar que para um mesmo conjunto de dados de entrada será obtido sempre um mesmo conjunto de resultados de saída.

No processo determinístico de alocação de tarefas, os parâmetros são definidos pela configuração inicial do problema de alocação, tais como, a proporção objetivo $\mathbb{P}$, o número de tarefas τ e robôs ρ. Ao garantir que os parâmetros não se alteram durante o tempo, o processo de alocação das tarefas entre os robôs será realizado sempre de uma mesma forma e obterá sempre o mesmo resultado final. Os processos determinísticos são previsíveis e, normalmente, mais simples que os processos estocásticos.

Na abordagem estocástica, o processo de decisão para a alocação de tarefas é baseado em eventos probabilísticos. Os eventos probabilísticos são representados por parâmetros aleatórios, que apresentam valores diferentes no decorrer do tempo. Estes eventos interagem com os parâmetros de entrada e produzem saídas aleatórias. Dessa forma, um processo estocástico de alocação que possua um conjunto conhecido de dados pode apresentar um conjunto imprevisível de resultados de saída, devido a interação dos elementos aleatórios no processo de decisão.

Os processos estocásticos de alocação de tarefas, são normalmente mais complexos que os processos determinísticos. Devido a característica imprevisível, podem produzir diferentes resultados de alocação durante o processo de busca pela alocação que respeite a proporção objetivo. Dessa forma, ao inserir o elemento aleatório, diferentes resultados de alocação são produzidos diminuindo as chances de que o processo de busca se restrinja a um conjunto de alocações localmente ótimos e alcance uma alocação que represente um resultado ótimo para o problema. Normalmente, retornam bons resultados de alocações, contudo, não é garantido que resultados de alocação ótimos sejam alcançados.

1.3.3 Procedimento Sequencial *vs* Procedimento Paralelo

No procedimento sequencial, a alocação de tarefas é dividida em ρ processos que são realizados sequencialmente pelos robôs. Cada processo é finalizado antes que o próximo seja iniciado, de modo que ao final de cada processo é realizada a alocação de uma tarefa para um dos robôs do enxame. Após a finalização do último processo, o enxame alcança a proporção desejada, conforme ilustrado na Figura 5.

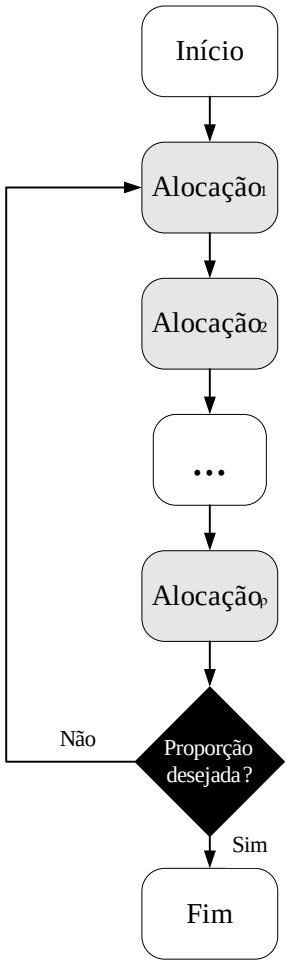


Figura 5: Estrutura da estratégia de alocação de tarefas sequencial

O algoritmo que utiliza a estratégia sequencial aloca cada tarefa a um robô de modo que apenas uma alocação seja realizada por vez. Dessa forma, para os casos em que não ocorrerem modificações na configuração inicial do enxame, tais como uma mudança no número de robôs, de tarefas ou na proporção objetivo durante o processo de alocação, o algoritmo converge garantidamente para uma solução. Contudo, a estratégia sequencial exige um elevado tempo de processamento para obter uma solução, uma vez que somente

após o término de realização do último processo o enxame alcançará a proporção.

No procedimento paralelo, a alocação de tarefas é também dividida em ρ processos, de modo que todos os processos são realizados simultaneamente, sendo um processo em cada robô. Ao final da realização do processo, o robô escolhe uma tarefa para ser realizada. Os processos são realizados pelos robôs do enxame de forma paralela onde os mesmos alocam suas tarefas baseando-se em informações obtidas através de troca de informação entre eles. Dessa forma, conforme as tarefas são escolhidas pelos robôs, a alocação de tarefas do enxame tende a convergir para a proporção desejada. A estrutura de realização das etapas segundo a abordagem paralela é ilustrada na Figura 6.

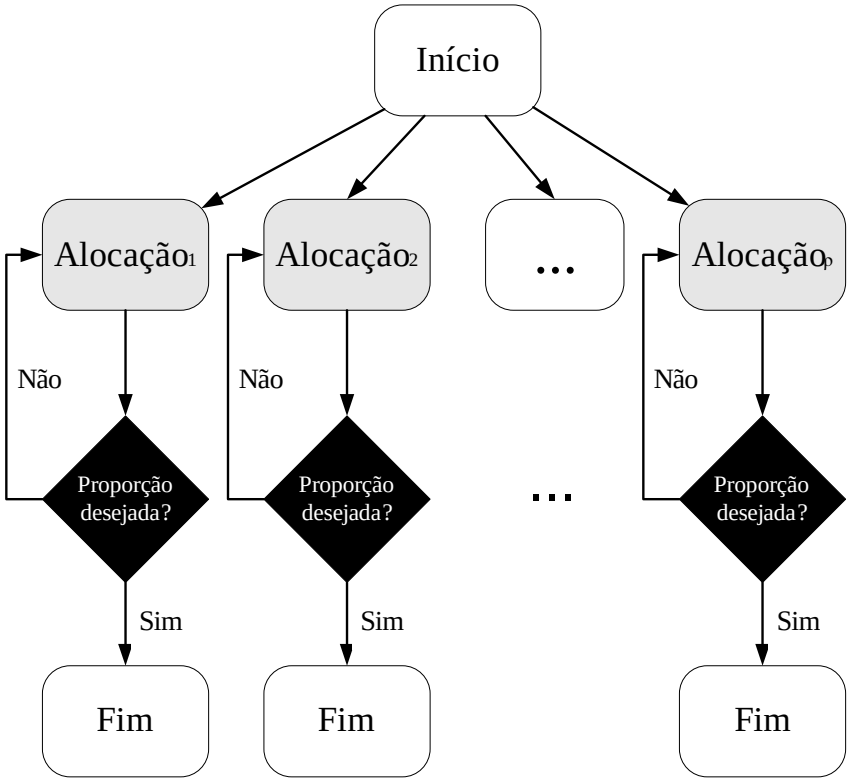


Figura 6: Estrutura da estratégia de alocação de tarefas paralela

Um exemplo de estratégia paralela é mostrado em (MENDONÇA; NEDJAH; MOURELLE, 2013). Neste algoritmo, os robôs trocam mensagens entre si de maneira contínua. Cada mensagem é composta pela identificação e a tarefa do robô emissor da mensagem. Dessa forma, o robô ao receber as mensagens dos demais atualiza uma tabela interna com os dados recebidos. Esta tabela dota o robô de um conhecimento sobre a alocação atual do enxame. A partir deste conhecimento o robô verifica a proporção do enxame

correspondente a esta alocação atual e escolhe sua tarefa com o intuito de convergir para a proporção objetivo desejada.

Note que neste algoritmo cada robô realiza a sua própria alocação de tarefa de forma distribuída. A escolha desta estratégia paralela de alocação de tarefas, torna o processo de decisão da alocação de tarefa concorrente entre os robôs. Esta escolha de estratégia agrega um menor tempo de convergência para a proporção desejada quando comparada a estratégia sequencial.

1.4 Considerações Finais do Capítulo

Este capítulo apresentou a definição do problema da alocação dinâmica de tarefas para grupos de robôs. O problema de alocação de tarefas é comparado a um problema de otimização, sendo o espaço de busca definido pelo número de alocações factíveis para o enxame. A complexidade do problema é influenciada pela composição do enxame, sendo os enxames homogêneos um problema de alocação de tarefas mais complexo quando comparados aos enxames heterogêneos.

O problema de ADT pode ser abordado de diferentes maneira. Pode explorar uma estratégia de controle centralizada ou distribuída, um processo de decisão determinístico ou estocástico ou mesmo um processo de execução sequencial ou paralelo.

Diversos estudos a respeito da alocação dinâmica de tarefas foram realizados nas últimas décadas. Diferentes metodologias foram propostas buscando realizar a alocação de tarefas de uma forma mais eficiente. O capítulo seguinte apresenta um estudo sobre os principais trabalhos realizados no estudo de algoritmos para a alocação de tarefas em grupos de robôs.

Capítulo 2

TRABALHOS RELACIONADOS

ESTE capítulo apresenta alguns dos algoritmos para alocação dinâmica de tarefas com destaque na comunidade acadêmica. Em geral são algoritmos desenvolvidos para sistemas distribuídos e com aplicação direta na área de robótica de enxame. Neste capítulo, nos limitaremos a tratar estritamente dos algoritmos formados por sistemas distribuídos em vista de sua flexibilidade e robustez ao ser empregado em problemas de alocação.

Recentemente, um certo número de soluções têm sido propostas na literatura para os problemas de alocação dinâmica de tarefas (ADT). Em (ZHANG; LIU, 2008) é realizada a separação dos algoritmos quanto a sua abordagem como comportamentais, baseados nas leis do mercado e os bio-inspirados. A Figura 7 ilustra a taxonomia.

A Seção 2.1 apresenta alguns dos algoritmos comportamentais de maior destaque. A Seção 2.2 apresenta alguns dos algoritmos inspirados nas Leis de Mercado. A Seção 2.3 apresenta alguns dos algoritmos inspirados nos comportamentos coletivos de inteligência de enxame. Na Seção 2.4 são realizadas as considerações finais para o capítulo.

2.1 Algoritmos Comportamentais

Os algoritmos baseados no comportamento são considerados robustos apresentando uma certa tolerância a falhas durante o processo de execução. Contudo a solução obtida pode apenas alcançar um ótimo local para o problema. Dentre os algoritmos comportamentais destacam-se o ALLIANCE (PARKER, 1998), BLE (WERGER; MATARIC, 2000) e ASyMTRe (TANG; PARKER, 2005a).

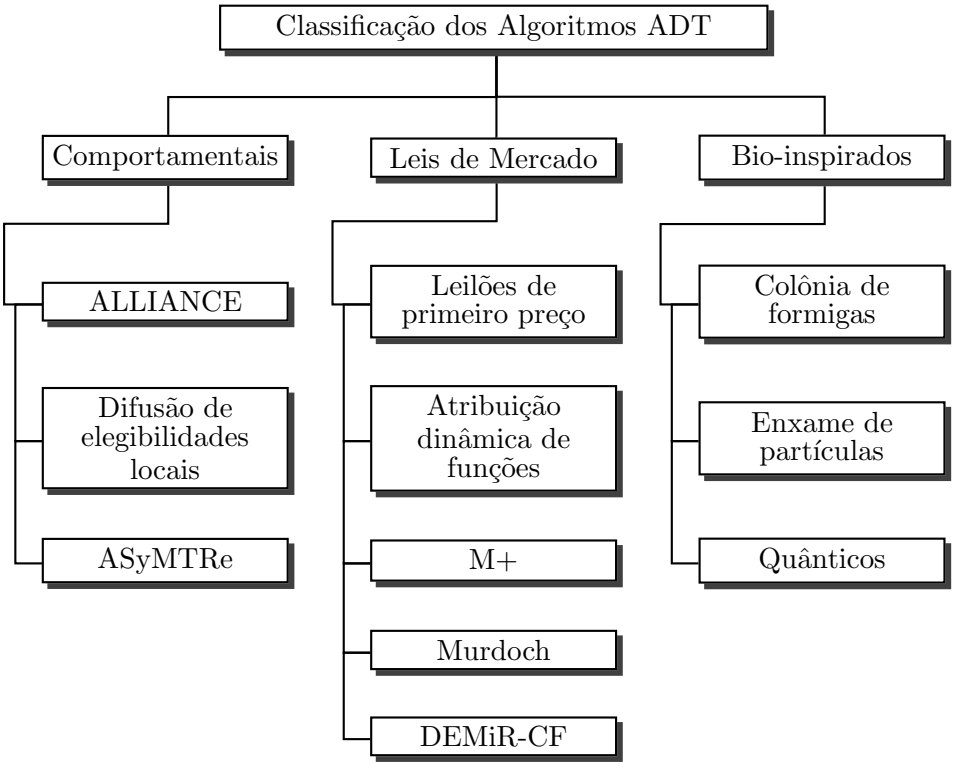


Figura 7: Classificação dos algoritmos ADT quanto a abordagem

2.1.1 ALLIANCE

O algoritmo ALLIANCE foi proposto em (PARKER, 1994) com o objetivo de estudar a cooperação tolerante a falhas dentro de grupos heterogêneos de robôs móveis. Sua utilização é indicada para grupos pequenos ou médios, ou seja, de 2 a 10 robôs. A arquitetura do algoritmo apresenta robustez ao adequar-se a situações de falhas e incertezas, tais como problemas de comunicação entre os robôs e retirada de indivíduos do grupo.

ALLIANCE é um algoritmo ST-SR-IA que utiliza intensivamente a comunicação tendo uma arquitetura totalmente distribuída que utiliza seleção de ação adaptativa para conseguir o controle cooperativo em missões envolvendo robôs de baixo interação e com tarefas independentes. Os robôs desta arquitetura possuem uma variedade de funções de alto nível para que possam desempenhar a missão devendo sempre escolher uma ação apropriada com base nos requisitos da missão, nas atividades realizadas pelos outros robôs, nas condições ambientais atuais, e em seus próprios estados internos.

Uma vez que tais grupos cooperativos muitas vezes trabalham em ambientes dinâmicos e imprevisíveis, a arquitetura permite que os membros do grupo respondam de

forma robusta e confiável às mudanças ambientais inesperados e modificações no grupo de robôs que podem ocorrer devido a uma falha mecânica, a aprendizagem de novas habilidades, ou a adição ou remoção de robôs da equipe por intervenção humana.

Sob a estrutura baseada em comportamento, as tarefas a serem realizadas são diferenciadas em grupos ditos *grupos comportamentais*. Por sua vez cada grupo comportamental é acionado baseado na motivação de cada robô em realizar determinada tarefa, ou seja, a motivação é designada a cada um dos robôs do enxame para realizar uma tarefa pelo tempo em que o mesmo mostrar possuir as melhores habilidades à realização daquela tarefa em comparação aos demais. A esta motivação é dado o nome de *comportamento motivacional*.

Cada robô recebe estímulos sensoriais que controlam alguns aspectos da ação executada por este. Os estímulos recebidos aliados ao grupo comportamental acionado definem o comportamento a ser realizado pelo robô em camadas de prioridade. Comportamentos de nível mais baixo, ou competências básicas do robô, correspondem a comportamentos primitivos de sobrevivência, tais como desvio de obstáculos. Enquanto que os comportamentos de nível superior correspondem a objetivos mais elevados, como a realização da tarefa. Os comportamentos de nível mais baixo podem ser suprimidos ou inibidos pelas camadas superiores, quando necessário através de um processo chamado *cross-inhibition*. Dessa forma, cada camada pode ser acionada produzindo o comportamento desejado.

Uma versão estendida desta arquitetura chamada L-ALLIANCE foi proposta diminuindo a dependência da comunicação no processo de alocação de tarefas. Nesta proposta, cada robô possui a habilidade adicional de atualizar seu conhecimento a respeito das habilidades que cada um dos demais possui em realizar certos tipos de tarefas. Esta arquitetura permite modelar o sistema de aprendizado humano de forma fácil e rápida agrupando a combinação apropriada de robôs para uma missão particular, uma vez que os robôs não precisam ter um conhecimento completo dos demais robôs do enxame.

2.1.2 Broadcast of Local Eligibility

O algoritmo Broadcast of Local Eligibility (BLE) é um algoritmo ST-SR-IA proposto para grupos heterogêneos de robôs. Possui o objetivo de criar grupos de cooperação para superar falhas permitindo inclusive a realocação das tarefas no enxame em caso falhas no

robô ou mesmo na impossibilidade do robô em executar uma tarefa atribuída à ele em decorrência de mudanças no ambiente.

É baseado na troca de mensagens para eleger a aptidão de cada robô em realizar cada tarefa. A aptidão do robô para realizar cada tarefa é definida por um valor numérico, ou nota, que cada robô gera através de um processo de auto-avaliação. O valor da aptidão é gerado baseando-se nas habilidades e dispositivos embarcados que o robô possui para realizar a tarefa. Quando a aptidão de um robô é a melhor para realizar algum comportamento, este envia uma mensagem de inibição de comportamento para os outros robôs, solicitando dessa forma esta tarefa para ele.

A comunicação é realizada pela troca de mensagens entre os robôs de forma balanceada devido a inserção do comportamento denominado *cross subsumption*. Este comportamento possui a função de instanciar os possíveis comportamentos do robô subordinado-os e arbitrando entre os comportamentos similares na escolha do comportamento de maior aptidão para este robô. A união dos comportamentos de elegibilidade e de subordinação permite ao enxame uma maior flexibilidade e escalabilidade para realizar uma missão composta por várias tarefas sem necessitar de uma negociação explícita ou de um conhecimento compartilhado de recursos.

2.1.3 ASyMTRe

O algoritmo Automated Synthesis of Multi-robot Task solutions through software Reconfiguration (ASyMTRe) é um algoritmo ST-MR-IA proposto para soluções com grupos heterogêneos de robôs. ASyMTRe automatiza o processo de geração de solução da tarefa, fornecendo a capacidade para os robôs colaborarem para encontrar novas soluções para as tarefas através de várias combinações de sensoriamento, atuadores e comportamentos que podem ser distribuídos através de múltiplos robôs. Dessa forma, uma cooperação próxima e dinâmica entre os membros do enxame para realizar as tarefas é alcançada.

A arquitetura proposta é fundamentada em quatro conceitos: programas de sensoriamento, programas de percepção, programas de movimentação e programas de comunicação, sendo este último o novo componente proposto. Estes programas são abstrações dos componentes que definem as capacidades individuais de cada robô, sejam eles físicos (sensores, atuadores) ou virtuais (processamento, armazenamento).

Os programas de sensoriamento tratam do controle de operação dos sensores e atuadores utilizados para que o robô perceba e interaja com as mudanças que ocorrem no ambiente ao seu redor. Os programas de percepção (MURPHY, 2000), processam os dados recebidos pelos sensores fornecendo informações necessárias para os programas de movimentação (ARKIN; BALCH; NITZ, 1993). Assim, os programas de percepção servem de interface entre os programas de sensoriamento que recebem os dados do ambiente transformando-os em informação relevante para os programas de movimentação que realiza ações físicas no robô, tais como a sua movimentação no ambiente.

Por fim, temos os programas de comunicação que realizam a comunicação entre os programas buscando uma interação entre os mesmo. O algoritmo proposto automatiza as conexões entre os programas buscando uma reconfiguração eficiente destas conexões para que o enxame alcance o seu objetivo.

O algoritmo troca mensagens entre os robôs ordenando-os em filas de prioridade para a execução das tarefas de acordo com a sua capacidade em executar a tarefa proposta. Os robôs menos capacitados recebem uma maior prioridade em relação aos demais com o objetivo de otimizar a eficiência individual de cada indivíduo no grupo. Dessa forma, caso o robô não possua todas as características necessárias para a realização desta tarefa é solicitada a participação de outros robôs até que a tarefa possa ser realizada por este conjunto de robôs. Ao término é esperado que sejam formados grupos de robôs capazes de realizar o conjunto de tarefas proposto.

ASyMTRe utiliza intensamente a comunicação através da troca de mensagens entre os robôs permitindo que as tarefas sejam realocadas no enxame adequando-se a situações de falha em ambientes dinâmicos. Uma análise formal de sua arquitetura é fornecida em (TANG; PARKER, 2005b).

2.2 Algoritmos Baseados nas Leis de Mercado

Os algoritmos baseados nas Leis de Mercado respeitam o preceito de maximizar as receitas (informação, rapidez) enquanto minimizam os custos (tempo de convergência, comunicação) com o objetivo de maximizar as vantagens de cada implementação.

A ideia foi proposta em (SMITH, 1980) através do método Contract Network Protocol (CNP). Devido a uma melhor escalabilidade, este método é particularmente bem

adequado para o domínio da robótica distribuída. Além disso, é garantido que sejam produzidas alocações ótimas, contudo os robôs devem colaborar através de uma comunicação explícita e com mais consumo dos recursos. Uma vez que a comunicação é interrompida, o desempenho deste método irá degradar significativamente (KALRA; MARTINOLI, 2006). Portanto, este é mais adequado para enxame de proporções pequena ou média.

Entre os algoritmos com destaque na atualidade podemos citar, First-price Auctions (ZLOT et al., 2002), Dynamic Role Assignment (CHAIMOWICZ; CAMPOS; KUMAR, 2002), M+ (BOTELHO; ALAMI, 1999), MURDOCH (GERKEY; MATARIĆ, 2000) e DEMiR-CF (SARIEL; BALCH, 2006).

2.2.1 First-price Auctions

O algoritmo First-price Auctions é um algoritmo ST-SR-IA proposto para soluções com grupos tanto homogêneos quanto heterogêneos de robôs. Sua abordagem é direcionada preferencialmente para a realização de tarefas de exploração de um ambiente desconhecido. Utiliza intensamente a comunicação através da troca de informações entre os robôs, contudo ainda apresenta funcionalidade mesmo que com reduzida eficiência para situações com baixa ou nenhuma comunicação.

A *receita* é paga para o robô em troca da informação do ambiente fornecida por ele e o *custo* é definido pela quantidade de recursos utilizados pelo robô na obtenção desta informação. Nesta implementação, o custo é representado pela distância média percorrida pelo robô para obter a informação e a receita é a informação propriamente dita calculada por sua relevância em comparação a um mapa de referência do ambiente.

O ambiente é representado por uma grade de ocupação dividida em células, onde cada célula pode receber a marcação de espaço livre, com obstáculo ou desconhecido. As informações obtidas por visitar um ponto de interesse podem ser calculada por meio da contagem do número de células desconhecidas contidas entre os pontos inicial e final. Assim, O lucro é calculado como a receita menos o custo gerados. Cada robô tenta maximizar a quantidade de novas informações descobertas, e minimizar a sua própria distância de deslocamento. Ao agir para promover os seus próprios interesses (maximizar o lucro), os robôs tendem a maximizar as informações obtidas por todo o grupo e minimizar o uso de recursos. Uma formalização da arquitetura baseada nas leis de mercado esta

disponível em (DIAS; STENTZ, 2000).

2.2.2 Dynamic Role Assignment

Dynamic Role Assignment é um algoritmo de classificação ST-SR-IA proposto preferencialmente para enxames heterogêneos de robôs, podendo também ser utilizado para um enxame homogêneo. Foi proposto baseado na cooperação distribuída que ocorre no *futebol de robôs*. Esta abordagem iniciou-se no final da década de 90 através da observação e estudo das *RoboCup Competitions* ((STONE; VELOSO, 1999), (CASTELPIETRA et al., 2000), (WEIGEL et al., 2001) e (EMERY; SIKORSKI; BALCH, 2002).

O algoritmo realiza a alocação das tarefas entre os robôs permitindo que ocorra uma realocação ou troca das tarefas entre os robôs. Este permite superar situações adversas, tais como, perda de comunicação e impossibilidade momentânea de realizar uma tarefa. Dessa forma, permitindo que os robôs troquem suas tarefas este algoritmo melhora a performance individual de cada robô e consequentemente de todo o enxame.

Em (CHAIMOWICZ; CAMPOS; KUMAR, 2002) é proposto uma variação deste algoritmo baseando-se em um sistema híbrido utilizando autômatos híbridos. O autômato híbrido é um autômato finito ampliado para um número finito de variáveis de valor real que mudam de forma contínua, conforme especificado por equações diferenciais (ALUR et al., 1995). Neles estão definidas as características individuais de cada robô. Através da troca de mensagens os robôs são capazes de sincronizar seus autômatos executando tarefas de forma cooperativa.

2.2.3 M+

O protocolo M+, classificado como ST-SR-IA, é baseado em uma combinação de planejamento local e negociação voltado para o problema da alocação de tarefas. Sua arquitetura permite a realocação de tarefas entre os robôs como fruto de uma reação cooperativa para a resolução de contingências locais, tais como falhas de comunicação, a decomposição de tarefas e o replanejamento das ações para a execução das tarefas.

O algoritmo foi concebido para ser integrado a arquitetura LAAS (ALAMI et al., 1998) e representa a primeira abordagem baseada no mercado para MRTA ou pelo menos o primeiro que foi motivado com o uso de leilões com ideias econômicas (GERKEY; MATARIĆ, 2004). Ele formaliza os recursos e os custos permitindo a realocação das tarefas apenas

quando uma nova tarefa torna-se disponível. Esta característica torna a aplicabilidade de M+ limitada para domínios onde os recursos e os custos são conhecidos.

Cada robô do enxame recebe uma mesma missão, sendo a missão um conjunto de tarefas. Por sua vez, as tarefas são definidas por um conjunto de metas a serem alcançadas. Dessa forma, através de um conhecimento local do enxame as tarefas são decompostas em séries de ações que quando executadas em uma dada sequência, definida pela decomposição, realizam a tarefa correspondente. As tarefas são atribuídas gradativamente através de um processo explícito de negociação entre os robôs combinado com um planejamento de atividades de estimativa de custos. O planejamento destas atividades permite a cada robô decidir sobre suas ações futuras, tendo em conta seu conhecimento local atual, as suas capacidades individuais e a dos demais robôs.

A arquitetura do algoritmo M+ é interpretada como uma etapa intermediária entre a primeira camada, que é a missão definida pelo problema, e a última, que representa a ação realizada por cada robô. Esta etapa é definida como a camada de tarefas. Ela pode ser composta por várias camadas, sendo cada uma definida por duas entidades: *planejadora* e *supervisora*. A entidade planejadora gera uma sequencia de ações necessárias para que que o objetivo desejado seja atingido enquanto a entidade supervisora interage com a camada seguinte, controlando a execução das ações e avaliando os possíveis novos eventos de entrada.

Nesse contexto, a entidade supervisora comunica-se com a planejadora da tarefa que está sendo realizada obtendo o plano com a sequência das ações necessárias à aquela tarefa. Assim a entidade supervisora utiliza o plano obtido para o cálculo do custo associado da tarefa servindo como base para o processo de negociação.

2.2.4 MURDOCH

MURDOCH é um algoritmo ST-SR-IA que baseia sua comunicação na expressiva troca de mensagens do tipo Publish/Subscribe (STROM et al., 1998). Este permite a análise do problema MRTA através de uma interface de mais alto nível para alocar tarefas a um grupo homogêneo ou heterogêneo de robôs. Em (GERKEY; MATARIC, 2002) é possível encontrar uma definição formal da arquitetura de MURDOCH.

A alocação de tarefas é realizada combinando o conjunto de recursos necessários

para executar a tarefa com os robôs que são capazes de realiza-la. Para isto, todos os robôs enviam aos demais uma mensagem contendo suas características, tais como os dispositivos que possuem (câmera, sonar), a mobilidade (móveis ou imóveis) entre outras. Por exemplo, um desktop enviaria uma mensagem com as características (imóvel, armazenamento de memória), enquanto um robô enviaria uma mensagem com as características (móvel, sonar, câmera, sensor infra-vermelho).

Através destas mensagens, cada robô se habilita para realizar uma dada tarefa comparando suas características com os recursos necessários para realiza-la. Nos casos em que mais de um robô habilita-se para uma mesma tarefa um processo de escolha do robô mais habilitado torna-se necessário. Dessa forma, os robôs se auto-avaliam retornado um valor referente ao seu fitness em realizar a tarefa baseando esta métrica nos conceitos das leis de mercado. O processo é repetido até que todas as tarefas sejam alocadas entre os robôs do enxame não permitindo uma realocação das mesmas. Assim, comparando-se MURDOCH com os outros algoritmos MRTA existentes, na ausência de novas tarefas à serem introduzidas e sem a possibilidade de re-atribuição de robôs que já tenham sido atribuídos, é impossível construir um melhor alocador de tarefas que MURDOCH (GERKEY; MATARIĆ, 2004).

2.2.5 DEMiR-CF

Distributed and Efficient Multi Robot - Cooperation Framework (DEMiR-CF) é um algoritmo classificado como MT-MR-IA. É proposto para grupos heterogêneos de robôs, sendo também utilizado para grupos homogêneos, realizarem missões complexas que incluam a existência de tarefas inter-relacionadas que exijam mais de um robô ou execuções simultâneas (SARIEL; BALCH; ERDOGAN, 2008).

O algoritmo é robusto respondendo a situações de contingência em tempo real de forma eficiente e mantendo o compromisso de apresentar uma alocação de tarefas distribuída de alta qualidade. Contudo, admite-se a possibilidade de apenas se obter resultados com ótimos locais para situações severas de contingência, como a perda de comunicação.

A estrutura do algoritmo combina a seleção das tarefas para os robôs mais capacitados e o uso da coalizão dos robôs em grupos para a execução da tarefa de forma

distribuída e organizada. Inicialmente, os robôs são informados sobre o conjunto de tarefas que compõem a missão a ser realizada e solicitam a execução das tarefas entre si considerando suas habilidades para esta tarefa. A cada solicitação feita é atribuído um custo global quanto a realização desta tarefa por este robô. A seguir, as solicitações passam por um processo de avaliação baseado em leilões que determina o robô mais capacitado para realizar a tarefa. Neste processo, inconsistências e conflitos são solucionados atribuindo cada tarefa para um robô mais capacitado ou em alguns casos para um grupo mais capacitado de robôs executarem esta tarefa.

Após atribuídas as tarefas o processo de avaliação sobre os robôs e as tarefas a eles atribuídas continua a ser realizado. Nesse processo são avaliadas novas informações recebidas do ambiente que podem ocasionar na troca da atribuição de uma tarefa para um outro robô desde que isto seja considerado mais rentável para a solução global. Dessa forma, é realizado um processo de avaliação de contingências em tempo real buscando adequar-se as novas condições impostas pelo ambiente no objetivo de alcançar uma boa solução global para o problema. Em (SARIEL, 2007) esta disponível uma abordagem formal sobre o algoritmo DEMiR-CF.

2.3 Algoritmos Bio-Inspirados

A utilização de algoritmos bio-inspirados para a resolução de problemas MRTA apresenta um considerável crescimento. Derivado dos comportamentos de insetos sociais, a abordagem da inteligência de enxame para este propósito apresenta características bastante apreciáveis, como a capacidade de auto-organização e comportamentos flexível as mudanças no ambiente através da simples interação entre os indivíduos.

Esta abordagem é apropriada para enxames homogêneos de robôs com capacidade limitada que através da interação alcançam resultados consideráveis e inalcançáveis para apenas um robô. Por sua característica limitada em muitos casos utilizam uma comunicação local implícita, através de sensores que monitoram o ambiente captando tanto excitações naturais (presença de obstáculos, mudança de luminosidade) como artificiais (causadas por outros robôs). Dessa forma, a medida que o número de robôs no enxame aumenta, a complexidade da comunicação cresce muito lentamente permitindo a realização de experimentos com um grande número de robôs, como mostrado em (ZHANG et al.,

2008) e (LIU; ZHANG, 2009).

Portanto, a abordagem bio-inspirada é a mais adequada para sistemas distribuídos multi-robô e por este motivo muitos pesquisadores têm desempenhado estudos nesta área. Em (YINGYING; YAN; JINGPING, 2003) e (YANG; WANG, 2004) foi realizada a cooperação em grupos de robôs utilizando-se a Otimização por Colônia de Formigas (ACO), enquanto que em (LIU et al., 2010) utilizou-se a Otimização por Enxame de Partículas (PSO). Em (YU et al., 2009) abordou-se os Algoritmos Evolucionários com Inspiração Quântica (QEA).

2.3.1 Algoritmo utilizando ACO

O algoritmo de otimização por colônia de formigas (ACO), proposto em (COLORNI et al., 1991), baseia-se no comportamento social de colônias de formigas que ao se movimentarem no ambiente depositam certa quantidade de uma substância chamada *feromônio*. Esta substância afeta o comportamento de outras formigas da forma que quanto maior for a quantidade desta substância em um determinado trajeto, maior será a probabilidade de outras formigas seguirem por este mesmo trajeto.

Em (YINGYING; YAN; JINGPING, 2003) utiliza-se ACO como ferramenta auxiliar no replanejamento de ações que objetivam a realização de uma tarefa. Neste, é ensaiado um experimento onde os robôs possuem a tarefa de deslocarem objetos de diferentes cargas de sua posição até uma região previamente definida. Considera-se que todos os robôs tenham conhecimento de sua localização e da localização desta região, sendo que cada robô possui a capacidade de deslocar objetos de até uma certa carga, exigindo que sejam acionados outros tantos robôs quanto necessários para que objetos de cargas maiores sejam deslocados.

Ao localizar um objeto o robô inicia o processo de realização da tarefa de deslocar este objeto até a região definida. Contudo, nas situações em que a carga do objeto for superior a sua capacidade de deslocamento o robô não terá sucesso em sua realização. Nestas situações, é estipulado um tempo limite para que a tarefa seja realizada. Quando este tempo limite é alcançado sem que a tarefa seja realizada o robô deposita uma certa quantidade de feromônio neste local o que irá atrair outros robôs. Dessa forma, através da formação coletiva de um grupo de robôs o objeto alvo é transportado para a região de destino.

Nesta implementação, de classificação ST-MR-IA, observa-se claramente a cooperação de um grupo de robôs para executar uma tarefa através de uma comunicação indireta, alocando e replanejando as tarefas entre seus membros de forma distribuída. Este tipo de comunicação aliado as características simples de cada robô permite uma melhor escalabilidade do enxame adaptando-se as necessidades de convergência impostas, tais como o tempo de convergência.

2.3.2 Algoritmo utilizando QEA

O Algoritmo Evolucionário com Inspiração Quântica (QEA) foi proposto em (HAN; KIM, 2002). Ele é baseado nos conceitos e princípios da computação quântica onde um bit quântico, diferente do bit clássico, pode possuir infinitos valores com características probabilísticas e não determinísticas. Na computação quântica existe um paralelismo intrínseco, uma vez que o estado quântico é uma superposição de estados básicos (SHOR, 1994).

Em (YU et al., 2009) e (YUPING; YINGHUA, 2007) é mostrada a utilização do algoritmo QACO, uma junção da inspiração quântica fornecida por QEA aliada ao algoritmo ACO. Nesta implementação a inspiração quântica é aplicada para solucionar problemas durante a formação de coalizão em grupos de robôs. A adição da parcela quântica possibilitou uma melhora nos resultados de otimização durante atividades de busca no ambiente mesmo para enxames pequenos de robôs.

2.3.3 Algoritmo utilizando PSO

A Otimização por Enxame de Partículas (PSO), proposta em (EBERHART; KENNEDY, 1995), é baseado no comportamento coletivo e no aprendizado social. No PSO, procura-se imitar o comportamento social de grupos de animais, mais especificamente de bando de pássaros. Nele os pássaros são vistos como partículas de um enxame onde o seu movimento é afetado por sua própria velocidade, a sua melhor posição e a melhor posição do enxame no passado. Como resultado, soluções ótimas podem ser obtidas em espaços de busca complexos.

Em (SALMAN; AHMAD; AL-MADANI, 2002) e (HO et al., 2008) é proposto o uso do PSO para a resolução do problema de alocação de tarefas em sistemas computacionais distribuídos. Nestas implementações o sistema procura distribuir a execução das tarefas

que compõem um programa entre os processadores com o objetivo de reduzir o tempo de execução do programa e para aumentar o rendimento do sistema.

Em (LIU et al., 2010) é proposto um algoritmo híbrido, de classificação ST-MR-IA, denominado PSACO que utiliza o PSO como ferramenta de otimização auxiliar no algoritmo ACO que realiza a alocação de tarefas em um enxame de robôs. O PSO busca otimizar o valor do feromônio a ser depositado por cada robô do enxame auxiliando a coalizão de grupos de robôs para a realização de tarefas.

2.4 Considerações Finais do Capítulo

Neste capítulo foram apresentados resumidamente diversos trabalhos sobre a temática de alocação de tarefas em um grupo de robôs. Os algoritmos presentes em cada um destes trabalhos foram apresentados destacando-se suas características particulares através de classificações formais e exemplos experimentais.Desta forma, foi estabelecida uma relação básica destacando-se as características positivas e negativas para cada implementação.

Ao fim, foi evidenciada a importância da abordagem dos algoritmos bio-inspirados na resolução de problemas MRTA sendo os mesmos os mais adequados para sistemas distribuídos em robôs móveis. O capítulo seguinte apresenta uma proposta de algoritmo determinístico para agir como solução ao problema MRTA.

Capítulo 3

ALGORITMO PARA ALOCAÇÃO DINÂMICA DE TAREFAS COM ABORDAGEM LOCAL

NESTE capítulo é proposto um algoritmo para a solução do problema de ADT baseando-se em uma abordagem local de alocação de tarefas. Na abordagem local, o robô realiza o ajuste de sua tarefa a partir de um processo determinístico de avaliação das tarefas realizadas pelos demais robôs do enxame. Os processos determinísticos são processos que apresentam um comportamento previsível, onde o resultado de cada execução é definido de maneira única.

Em geral, o algoritmo é organizado em 3 etapas principais: *atualização*, *ajuste* e *execução*. A Seção 3.1 deste capítulo apresenta uma visão geral do algoritmo. A Seção 3.2 é dedicada à etapa de atualização, enquanto a Seção 4.4 descreve os detalhes da etapa de ajuste da tarefa. Na Seção 3.4 são apresentadas as considerações finais deste capítulo.

3.1 Visão geral do algoritmo

O Algoritmo para Alocação Dinâmica de Tarefas com abordagem Local (ADTL) proposto define dinamicamente a alocação da tarefa a partir de um processo de ajuste. ADTL é estruturado em etapas bem definidas, conforme mostra o Algoritmo 1.

As etapas são repetidas iterativamente até atingir a proporção desejada. Cada robô do enxame executa localmente uma versão deste algoritmo. Neste contexto, o robô é responsável por definir dinamicamente a atribuição da tarefa a executar utilizando um processo de avaliação. Durante o processo, é definida a tarefa a ser alocada à ele baseado-

se no conhecimento atual, mesmo que parcial, das tarefas atribuídas aos demais robôs do enxame.

Algoritmo 1 permite alocar τ tarefas a um enxame de ρ robôs, onde a proporção objetivo é $\mathbb{P}$. Note que t_i é o identificador da tarefa atualmente alocada ao robô id_i e $\mathbb{A}_i$ a alocação de tarefas do ponto de vista deste robô. Inicialmente, $\mathbb{A}_i$ é configurada com uma alocação inicial $\mathbb{A}_0$ (linha 1). A inicialização da alocação inicial pode ser realizada de forma randômica, onde a alocação de tarefa para cada robô é obtida aleatoriamente a partir das tarefas válidas definidas por $\mathbb{T}$, ou fixa, onde a alocação de tarefa para cada robô é previamente definida.

Algoritmo 1 ADTL no robô i

Entrada τ, ρ, $\mathbb{P}$ e $\mathbb{A}_0$;
1: $\mathbb{A}_i := \mathbb{A}_0$;
2: **Enquanto** *verdadeiro* **Faça**
3: Atualizar($\mathbb{A}_i$);
4: **Se** $\mathbb{P} \neq \mathbb{P}_i$ **Então**
5: AjustarTarefa($\mathbb{A}_i, t_i$);
6: **Senão**
7: ExecutarTarefa(t_i) por um determinado período de tempo;
8: **Fim Se**
9: **Fim Enquanto**;

A cada iteração é realizada a atualização de $\mathbb{A}_i$ (linha 3), conforme detalhado na Seção 3.2. Em seguida, é verificado se a proporção desejada $\mathbb{P}$ é alcançada (linha 4). Comparando-a à proporção atual $\mathbb{P}_i$, que é obtida a partir da alocação $\mathbb{A}_i$ conforme explicado no Capítulo 1. No caso negativo, é realizado o ajuste da tarefa t_i (linha 5). O procedimento utilizado para ajustar a tarefa do robô é explicado na Seção 4.4. Caso a proporção desejada tenha sido alcançada, a tarefa t_i é realizada (linha 7).

A etapa de execução, implementada por $ExecutarTarefa$, realiza a tarefa t_i (linha 7) alocada para o robô de acordo com $\mathbb{A}_i$. Em geral, as τ tarefas consideradas podem possuir características distintas em relação à ação que deve ser executada e ao tempo de execução desta ação, sendo estas características definidas caso a caso dependendo da implementação e do problema.

A dinâmica imposta no Algoritmo 1 permite que o robô repita as etapas de atualização e ajuste várias vezes, objetivando alcançar uma distribuição das tarefas de acordo com a proporção $\mathbb{P}$ prescrita. Com essa distribuição de tarefas, os robôs do enxame pas-

sam a realizar a tarefa correspondente até que uma nova configuração do problema é informada. Vale lembrar que os parâmetros são o número total de robôs ρ, o número total de diferentes tarefas a serem realizadas τ e a proporção desejada $\mathbb{P}$.

3.2 Etapa de Atualização

A etapa de atualização é implementada conforme mostrado no Algoritmo 2. Nesta etapa é realizada a troca de informação entre os robôs e atualização do conhecimento deste em relação às tarefas alocadas aos demais robôs do enxame. Dessa forma, o robô i envia uma mensagem, composta pelo seu identificador id_i e por sua tarefa t_i atualmente alocada, aos demais robôs do enxame com o objetivo de informá-los sobre a sua alocação atual.

Algoritmo 2 Atualizar no robô i

Entrada τ, ρ, $\mathbb{A}_i$ e $\mathbb{I}$;
Saída $\mathbb{A}_i$, $\mathbb{P}_i$;
1: $msg \leftarrow \langle id_i, t_i \rangle$;
2: **Enviar** msg aos demais $\rho - 1$ robôs;
3: **Receber** msg dos demais $\rho - 1$ robôs;
4: **Para** cada mensagem $\langle id_j, t_j \rangle$ recebida **Faça**
5: $\quad \mathbb{A}[\mathbb{I}[j]] := t_j$;
6: **Fim Para**
7: **Para** $r := 1 \rightarrow \rho$ **Faça**
8: $\quad \mathbb{C}_i[\mathbb{A}_i[r]] := \mathbb{C}_i[\mathbb{A}_i[r]] + 1$;
9: **Fim Para**
10: **Para** $t := 1 \rightarrow \tau$ **Faça**
11: $\quad \mathbb{P}_i[t] := \mathbb{C}_i[t]/\rho$;
12: **Fim Para**

Cada robô possui um conhecimento próprio a respeito das tarefas atualmente alocadas aos demais robôs. Note que este conhecimento pode ser diferente de um robô a outro devido a perda de mensagens durante a fase de troca de informações entre os robôs. Assim através da troca de mensagens entre os robôs, onde os mesmos informam o seu identificador e a sua respectiva tarefa, o robô consegue ter um conhecimento, mesmo que parcial, sobre a alocação de tarefas dos demais robôs do enxame. A atualização da alocação de tarefas $\mathbb{A}_i$ depende do recebimento de mensagens dos demais robôs. Devido a isto, $\mathbb{A}_i$ varia continuamente no tempo levando em conta os momentos distintos em que os robôs recebem as mensagens que lhes são destinadas.

De maneira complementar, o robô recebe dos demais robôs mensagens de mesmo formato, sendo informado sobre as tarefas alocadas aos mesmos. Baseando-se nestas mensagens recebidas o robô atualiza $\mathbb{A}_i$ com as tarefas dos demais robôs (linha 5).

Uma vez a alocação $\mathbb{A}_i$ atualizada, o robô calcula a nova proporção do enxame $\mathbb{P}_i$. Com este objetivo, é contabilizado o número de robôs alocados a cada uma das tarefas em τ contadores, baseando-se na nova alocação $\mathbb{A}_i$ (linhas 7 - 9). O conjunto dos contadores é representado por $\mathbb{C}_i$, onde $\mathbb{C}_i = \{c_1, c_2, \ldots, c_\tau\}$ e c_i representa a quantidade de robôs atualmente associados à tarefa de identificador i. Em seguida, é realizada a atualização de $\mathbb{P}_i$ através do cálculo das proporções equivalentes a cada uma das tarefas em relação ao número de robôs ρ (linhas 10 - 12), conforme definido na Equação 1 e na Equação 2 do Capítulo 1.

3.3 Etapa de Ajuste

O Algoritmo 3 descreve o método proposto para implementar o ajuste das tarefas alocadas aos robôs do enxame para alcançar a proporção objetivo. Esta etapa é responsável por decidir se o robô deve ou não alterar a alocação de sua tarefa, e caso positivo, para qual nova tarefa deverá mudar. Vale lembrar que o robô i está associado a tarefa t_i e tem $\mathbb{A}_i$ como alocação atual.

Inicialmente, é realizada a comparação entre o número de robôs atualmente alocados a tarefa t_i dado por $\mathbb{C}_i[t_i]$ e o número de robôs que devem estar alocados a esta tarefa $\mathbb{C}[t_i]$ para que a proporção objetivo $\mathbb{P}[t_i]$ para esta tarefa seja alcançada. Nesta comparação, três situações são possíveis:

1. $\mathbb{C}_i[t_i] > \mathbb{C}[t_i]$, indicando que existe um excesso de robôs alocados para a tarefa t_i (linha 1 do Algoritmo 3);

2. $\mathbb{C}_i[t_i] = \mathbb{C}[t_i]$, indicando que existe um equilíbrio entre o número desejado de robôs alocados e o número real de robôs que estão alocados para a tarefa t_i;

3. $\mathbb{C}_i[t_i] < \mathbb{C}[t_i]$, indicando que existe uma falta de robôs alocados para a tarefa t_i.

Para a situação onde tem um excesso de robôs atualmente alocados para a tarefa do robô (t_i), o robô provavelmente deve mudar de tarefa. Se for o caso, o processo de

Algoritmo 3 AjustarTarefa no robô i

Entrada τ, ρ, $\mathbb{A}_i$, t_i, $\mathbb{I}$, $\mathbb{C}$ e $\mathbb{C}_i$;
Saída t_i;

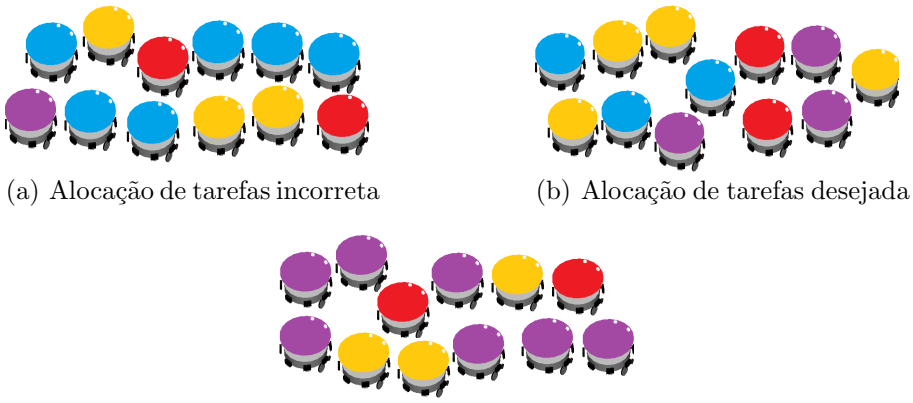
```
1: Se ℂ_i[t_i] > ℂ[t_i] Então
2:    Para t := 1 → τ Faça
3:       Se ℂ_i[t] < ℂ[t] Então
4:          δ[t] := ℂ[t] − ℂ_i[t];
5:       Senão
6:          δ[t] := 0;
7:       Fim Se
8:    Fim Para
9:    t' := t_k|δ[t_k] é o maior, k = 1...τ;
10:   NR := 0;
11:   Para r := 1 → ρ Faça
12:      Se 𝕀[r] ≠ id_i Então
13:         t_r := 𝔸_i[𝕀[r]];
14:         Se 𝔸_i[r] ≠ t' e ℂ_i[t_r] > ℂ[t_r] e 𝕀[r] < id_i Então
15:            NR := NR + 1;
16:         Fim Se
17:      Fim Se
18:   Fim Para
19:   Se δ[t'] > NR Então
20:      t_i := t'
21:   Fim Se
22: Fim Se
```

decisão precisa ainda determinar para qual tarefa o robô deverá mudar sua alocação. Para as demais situações, o processo de decisão é encerrado e o robô mantém a sua alocação atual de tarefa.

Confirmada a possibilidade de alteração de tarefa, é iniciado o processo que busca definir a nova alocação de tarefa mais apropriada para o robô. Para isto, são identificas as tarefas que possuem uma falta de robôs associados a elas, como tarefas promissoras a serem usadas durante a escolha da nova alocação do robô. Note que estas tarefas são identificadas por um resultado positivo na diferença $\delta[t]$, sendo t o identificador de uma das tarefas.

A ação de identificar as tarefas promissoras baseia-se no compromisso de reduzir a diferença entre os contadores $\mathbb{C}[t]$ e $\mathbb{C}_i[t]$, sendo t o identificador de uma das tarefas. Note que caso todas as diferenças forem nulas, ou seja, $\delta[u] = 0$ para $u = 1 \ldots \tau$, pela percepção do robô i, então está alcançada a proporção objetivo, uma vez que têm-se $\mathbb{P} = \mathbb{P}_i$.

As tarefas promissoras a serem usadas durante a escolha da nova alocação de



(a) Alocação de tarefas incorreta (b) Alocação de tarefas desejada

(c) Alocação de tarefas incorreta

Figura 8: Ilustração de alocação de tarefas para um grupo de 12 robôs

tarefa do robô são ordenadas em termos de suas diferenças δ. A tarefa t' que apresentar a maior diferença $\delta[t']$ é selecionada (linha 9). Nos casos em que ocorrer um empate entre duas ou mais diferenças, é escolhida a tarefa com o maior identificador como sendo a tarefa promissora a ser a possível nova tarefa do robô, unicamente por uma questão de desempate. É importante mencionar, que ao selecionar a nova tarefa para o robô não está necessariamente garantido que ao término do processo de ajuste o robô irá alterar sua alocação atual para esta nova tarefa. Isto é, caso exista necessidade da alteração, a tarefa assim selecionada será a nova alocação de tarefa do robô.

Para avaliar a confirmação de mudança na alocação, são observadas as condições dos demais robôs que estão também com possibilidade de mudança de alocação. Dessa forma, antes de tomar a decisão de mudar a alocação, uma avaliação prévia levando em conta a decisão dos demais robôs é realizada. Isto permite evitar que decisões iguais, além do necessário, ocorram simultaneamente em diferentes robôs invalidando a convergência do processo. O número máximo de robôs que podem tomar a mesma decisão é calculado previamente (linhas 11 – 18). A seguir, é ilustrada a situação através de um exemplo.

Seja o caso mostrado na Figura 8. Neste exemplo, o enxame é composto de 12 robôs, sendo permitidas 4 tarefas: t_A, t_B, t_C e t_D. Na Figura 8, cada tarefa é representada por uma cor, sendo a cor azul correspondente a tarefa t_A, a cor amarela à tarefa t_B, a cor vermelha à tarefa t_C e a cor roxa à tarefa t_D. Suponha que a proporção objetivo seja $\mathbb{P} = \{\frac{3}{12}, \frac{4}{12}, \frac{2}{12}, \frac{3}{12}\}$, conforme mostrada em Figura 8(b), ou seja, 3 robôs alocados à tarefa t_A, 4 robôs à tarefa t_B, 2 robôs à tarefa t_C e por fim 3 robôs à tarefa t_D

($\mathbb{C} = \{3, 4, 2, 3\}$). Dessa forma, para a alocação de proporção atual $\mathbb{P}_a = \{\frac{6}{12}, \frac{3}{12}, \frac{2}{12}, \frac{1}{12}\}$, sendo $\mathbb{C}_a = \{6, 3, 2, 1\}$, mostrada em Figura 8(a) teríamos um excesso de 3 robôs alocados à tarefa t_A, que de acordo com o ajuste de alocação, deveriam ser alocados entre as demais tarefas. Nesta situação, a tarefa t' selecionada pelo processo de ajuste seria t_D, devido a mesma apresentar a maior diferença $\delta[t_D] = 2$ entre as tarefas, caracterizando que esta tarefa apresenta o maior número de robôs faltando para que se atinja o equilíbrio ($\delta[t_D] = 0$).

Assim os 6 robôs atualmente alocados à tarefa t_A escolheriam como uma nova possível alocação a tarefa t_D. Deste modo, caso não seja definida uma prioridade à mudança de alocação entre os robôs predispostos para efetuá-las, existe a possibilidade de mais de 2 robôs alterarem suas alocações para t_D, tornado-a uma tarefa com excesso de robôs alocados. No exemplo proposto, no pior caso, todos os 6 robôs atualmente alocados à tarefa t_A alteraram suas alocações para a tarefa t_D, como mostra a Figura 8(c), definindo a nova proporção $\mathbb{P}_c = \{\frac{0}{12}, \frac{3}{12}, \frac{2}{12}, \frac{7}{12}\}$, sendo $\mathbb{C}_c = \{0, 3, 2, 7\}$. Em decorrência surge uma situação inversa a original em que a tarefa t_A passaria a apresentar uma falta de robôs alocados e a tarefa t_D um excesso. Esta situação é um empecilho ao processo de convergência pois impede que o enxame altere a sua proporção atual no sentido de convergir para a proporção objetivo.

Para remediar este comportamento, ao robô com o identificador de valor mais baixo, dentre os robôs predispostos à mudança de alocação, é dada maior prioridade para realizar a alteração. Seguindo este preceito, para o robô id_i e de acordo com o seu conhecimento $\mathbb{A}_i$ da alocação atual do enxame, é contabilizado o número de robôs predispostos a alterar sua alocação que possuam identificador menor que id_i. Esta quantidade de robôs, contabilizada em NR do Algoritmo 3, refere-se ao número de robôs com maior prioridade à mudança de alocação em relação ao robô id_i. Desta forma, caso o número de robôs necessários para que a nova alocação atinja o equilíbrio, i.e. NR seja maior do que o número de robôs com prioridade à mudança em relação a este, o robô id_i irá confirmar a alteração de sua alocação para a tarefa t'. Caso contrário, o robô irá manter a sua tarefa como t_i.

3.4 Considerações Finais do Capítulo

Neste capítulo foi apresentado o algoritmo com abordagem local de alocação dinâmica de tarefas ADTL desenvolvido como uma solução ao problema de alocação de tarefas em grupos de robôs. ADTL é um algoritmo distribuído estruturado em três etapas. No capítulo 5 serão apresentadas as implementações deste algoritmo em software e em robôs móveis reais. O capítulo seguinte apresenta a descrição do algoritmo com abordagem global proposto como solução ao problema de alocação de tarefas em grupos de robôs.

Capítulo 4

ALGORITMO PARA ALOCAÇÃO DINÂMICA DE TAREFAS COM ABORDAGEM GLOBAL

NESTE capítulo é proposto um algoritmo para a solução do problema ADT baseando-se em uma abordagem global de alocação de tarefas (ADTG). Nesta abordagem, o robô realiza o ajuste de sua alocação de tarefas a partir de um processo estocástico. Os processos estocásticos são processos não determinísticos, baseando-se em eventos aleatórios. O algoritmo proposto explora a otimização por enxame de partículas (*Particle Swarm Optimization* – PSO).

De modo geral, o algoritmo é estruturado em várias etapas que objetivam a minimização de uma função objetivo, permitindo a convergência do enxame para a proporção desejada. O algoritmo é executado de forma distribuída em cada robô do enxame, definindo dinamicamente a alocação da tarefa a ser realizada por cada robô.

A Seção 4.1 apresenta uma introdução à otimização por enxame de partículas. A Seção 4.2 fornece uma visão geral da dinâmica de execução do algoritmo proposto. A Seção 4.3 e Seção 4.4 detalham as etapas do algoritmo ADTG proposto. A Seção 4.5 apresenta as considerações finais para o capítulo.

4.1 Introdução ao PSO

O PSO é um algoritmo estocástico baseado em inteligência de enxame. O algoritmo procura a solução de problemas de otimização em um determinado espaço de busca de forma iterativa. Foi inicialmente desenvolvido como uma ferramenta de simulação de

padrões de voo dos pássaros em busca de comida e proteção (REYNOLDS, 1987). O algoritmo de PSO foi proposto por Kennedy e Eberhart em (KENNKDY; KBEHHART, 1995), que observaram a possibilidade de adaptar o comportamento social dos pássaros para ser utilizado em processos de otimização, criando assim a primeira versão do PSO.

O PSO gerencia um conjunto de partículas, onde cada partícula representa uma solução em potencial para o problema. Geralmente, o problema é representado por uma função objetivo. Cada partícula possui uma posição, velocidade e direção adaptativas que determinam sua movimentação no espaço de busca. Durante cada iteração do algoritmo, a partícula se locomove pelo espaço de busca, tendo sua posição ajustada de acordo com a sua própria experiência e com a experiência das partículas vizinhas. A atualização da posição da partícula é realizada aplicando-se uma velocidade para cada dimensão do espaço de busca (ENGELBRECHT, 2005).

Em (KENNKDY; KBEHHART, 1995) é proposto o algoritmo de otimização denominado de *Global Best PSO*. Este algoritmo representa uma versão do PSO com uma ampla noção de vizinhança, onde a vizinhança de cada partícula é formada por todas as demais partículas do enxame.

4.1.1 Global Best PSO

O algoritmo *Global Best* PSO (GBPSO) considera o melhor resultado global do enxame a cada iteração. A estrutura do GBPSO é apresentada no Algoritmo 4, onde NP denota o número de partículas do enxame e ND o número de dimensões do problema. Cada partícula i possui uma posição x_i e velocidade v_i. A posição e velocidade de uma partícula são definidas para cada dimensão do problema.

O algoritmo PSO inicializa a posição da partícula no espaço de busca de forma aleatória. Após a inicialização, um processo iterativo é iniciado, onde a aptidão de cada partícula é calculada. A aptidão de uma partícula é determinada pelo valor da função objetivo associada na posição da partícula. Em seguida, são comparados os valores de aptidão da partícula na sua posição atual que ocasionou o melhor valor de aptidão durante seu passado. Este valor é denominado *Pbest*. A melhor posição associado ao *Pbest* da partícula é atualizada sempre que o valor de aptidão da partícula é menor ou igual do que seu *Pbest*. Também é realizada a comparação do *Pbest* da partícula com o melhor valor de

Algoritmo 4 Algoritmo GB – PSO

```
Para i = 1 → NP Faça
  inicializar x_i aleatoriamente;
Fim Para;
y_i ← f(x_i);
ŷ ← y_ii;
Repita
  Para i = 1 → NP Faça
    Se f(x_i) ≤ f(y_i) Então
      y_i ← x_i;
    Fim Se;
    Se f(y_i) ≤ f(ŷ) Então
      ŷ ← y_i;
    Fim Se;
    Para j = 1 → ND Faça
      Atualizar a velocidade da partícula i na dimensão j (v_ij) utilizando a Equação 9;
      Atualizar a coordenada j da posição da partícula i (x_ij) utilizando a Equação 10;
    Fim Para
  Fim Para;
Até Critério de parada;
```

aptidão *Pbest* entre todas as partículas para obter o melhor de todos, denominado *Gbest*. A melhor posição global do enxame é atualizada sempre que o *Pbest* de uma partícula é menor ou igual ao *Gbest*.

Depois de atualizar a melhor posição do enxame, o algoritmo realiza a atualização das velocidades referentes a cada partícula. As velocidades da partícula são calculadas de acordo com a Equação 9, sendo $v_{ij}(t+1)$ a velocidade atualizada de $v_{ij}(t)$ da partícula i na dimensão j:

$$v_{ij}(t+1) = wv_{ij}(t) + c_1 r_{1j}(t)\left(y_{ij} - x_{ij}(t)\right) + c_2 r_{2j}(t)\left(\hat{y}_j - x_{ij}(t)\right), \tag{9}$$

onde w é chamado de *coeficiente de inércia*, r_{1j} e r_{2j} são valores aleatórios do intervalo [0,1) gerados cada iteração, c_1 e c_2 são constantes positivas, y_{ij} é a coordenada da melhor posição atingida pela partícula i, na dimensão j, no passado, $\hat{y}_j$ é a coordenada da melhor posição na dimensão j, atingida no passado, entre todas as partículas e $x_{ij}(t)$ e $v_{ij}(t)$ são a posição e velocidade atuais da partícula i na dimensão j. Dessa forma, a próxima velocidade de uma partícula é definida por três termos:

- $wv_{ij}(t)$: o componente *de inércia*, que previne a partícula de mudanças drásticas em termos de direção;

- $c_1 r_{1j}(y_{ij} - x_{ij}(t))$: o componente *cognitivo*, que possui o efeito de atrair a partícula para sua melhor posição referente a *Pbest* do passado;

- $c_2 r_{2j}(\hat{y}_j - x_{ij}(t))$: o componente *social*, que possui o efeito de atrair a partícula para a melhor posição referente a *Gbest* encontrada por todas as partículas.

A posição da partícula é atualizada de acordo com a Equação 10,

$$x_{ij}(t+1) = v_{ij}(t+1) + x_{ij}(t), \tag{10}$$

onde $x_{ij}(t+1)$ representa a próxima posição e $x_{ij}(t)$ a coordenada de posição atual da partícula i na dimensão j. Após a atualização das velocidades e da posição de cada partícula é verificado o critério de parada. Sendo o critério de parada alcançado, é registrado o resultado final, e caso contrário é realizada mais uma iteração.

4.1.2 Parâmetros do GBPSO

Cada parâmetro do algoritmo GB PSO está intimamente relacionado à atualização da velocidade e posição da partícula. Dessa forma, a escolha do valor de cada parâmetro é fundamental no processo de otimização.

O coeficiente de inércia w controla a relação entre a ação de explorar grandes áreas ou uma determinada localidade pequena do espaço de busca. Os coeficientes cognitivo c_1 e social c_2 definem a importância da experiência da partícula e do enxame, respectivamente, no cálculo da velocidade da partícula. Os fatores r_1 e r_2 definem a parcela estocástica das contribuições cognitiva e social da partícula.

O tamanho do enxame define a capacidade de se abranger uma determinada porção do espaço de busca em cada iteração do algoritmo. O espaço de busca é definido pelo conjunto das possíveis soluções. O PSO é comumente utilizado para aplicações de otimização em espaços de busca contínuos. Contudo, para algumas implementações é proposta uma abordagem discreta deste espaço de busca.

O critério de parada é utilizado pelo algoritmo para decidir quanto ao término do processo de otimização. É importante que o critério de parada utilizado não implique

em interromper o processo de busca prematuramente, antes de se obter um resultado satisfatório, ou levar a um processamento desnecessário quando o resultado já tiver sido alcançado.

4.1.3 GBPSO Paralelo

O PSO na sua forma original é inspirado em comportamentos e aprendizados coletivos de indivíduos. Dessa forma, possui uma estrutura altamente paralelizável. O processamento paralelo é uma estratégia utilizada para acelerar a resolução de problemas computacionais complexos, dividindo-os em processos que são executados simultaneamente. Em geral, a execução desses processos requer uma comunicação entre os mesmos com o objetivo de sincronização ou troca de informações.

Como proposta de paralelização do GBPSO, foi utilizado o algoritmo denominado PPSO (*Parallel PSO*) (**??**). O PPSO foi idealizado considerando o fato de que o trabalho realizado por uma partícula, na sua maioria, é independente daquele realizado pelas demais partículas do enxame. A Figura 9 apresenta o fluxograma do algoritmo PPSO, onde $v_1, \ldots, v_{NP}$, $x_1, \ldots, x_{NP}$ e $y_1, \ldots, y_{NP}$ representam, as velocidades, posições e melhores posições para as NP partículas do enxame, respectivamente, e $\hat{y}$ a melhor posição do enxame.

A Figura 9 apresenta a paralelização das partículas do enxame através da execução de NP processos paralelos, sendo um para cada partícula. Cada processo realiza a inicialização da velocidade v_i e posição x_i da partícula i. Após a inicialização, cada processo calcula o valor de aptidão através da função objetivo f e elege *Pbest* de forma independente e em paralelo com os demais processos. No entanto, uma comunicação é necessária entre todas as partículas do enxame, para trocar o valor *Pbest*, permitindo a eleição de *Gbest*. Com o objetivo de sincronizar o processo, a atualização da velocidade e posição é iniciada somente após esta etapa de comunicação. Em seguida, é realizada a verificação do critério de parada, de modo que caso o mesmo seja alcançado, o algoritmo é encerrado. Caso contrário, é realizada uma nova iteração do algoritmo. Note que a atualização do *Gbest* e a verificação do critério de parada são realizados por um único processo pré-definido.

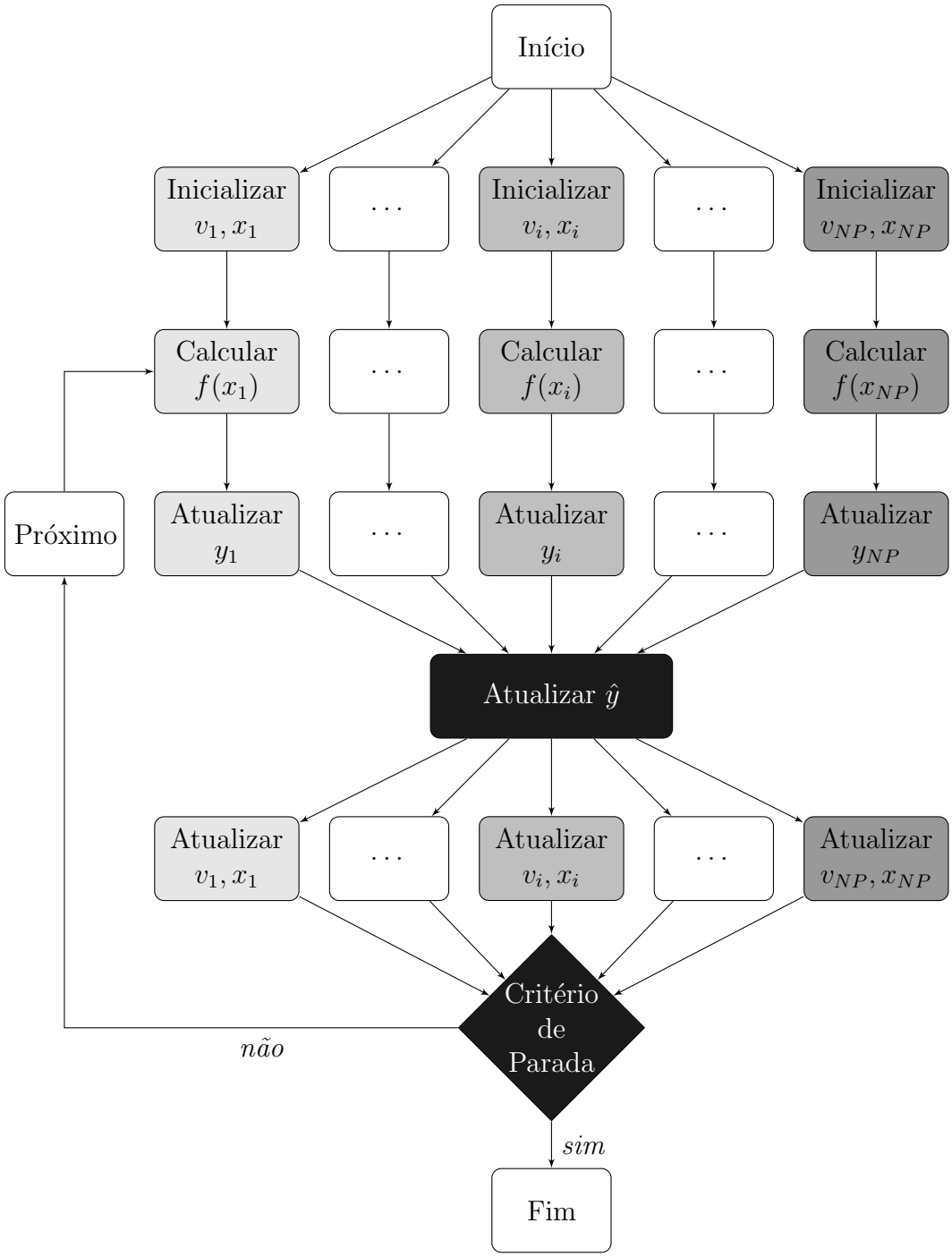


Figura 9: Computação paralela realizada pelo PPSO para um exame com NP partículas

4.2 Visão Geral da ADTG

O Algoritmo para Alocação Dinâmica de Tarefas com abordagem Global (ADTG) proposto define, de forma dinâmica, o ajuste da alocação de tarefas $\mathbb{A}_i$ do ponto de vista do robô id_i. O algoritmo baseia-se no processo de otimização do algoritmo GBPSO como forma de minimizar a função objetivo idealizada para o problema de ADT. Sua represen-

tação é mostrada na Figura 10 através da execução de NP processos paralelos, sendo um para cada partícula.

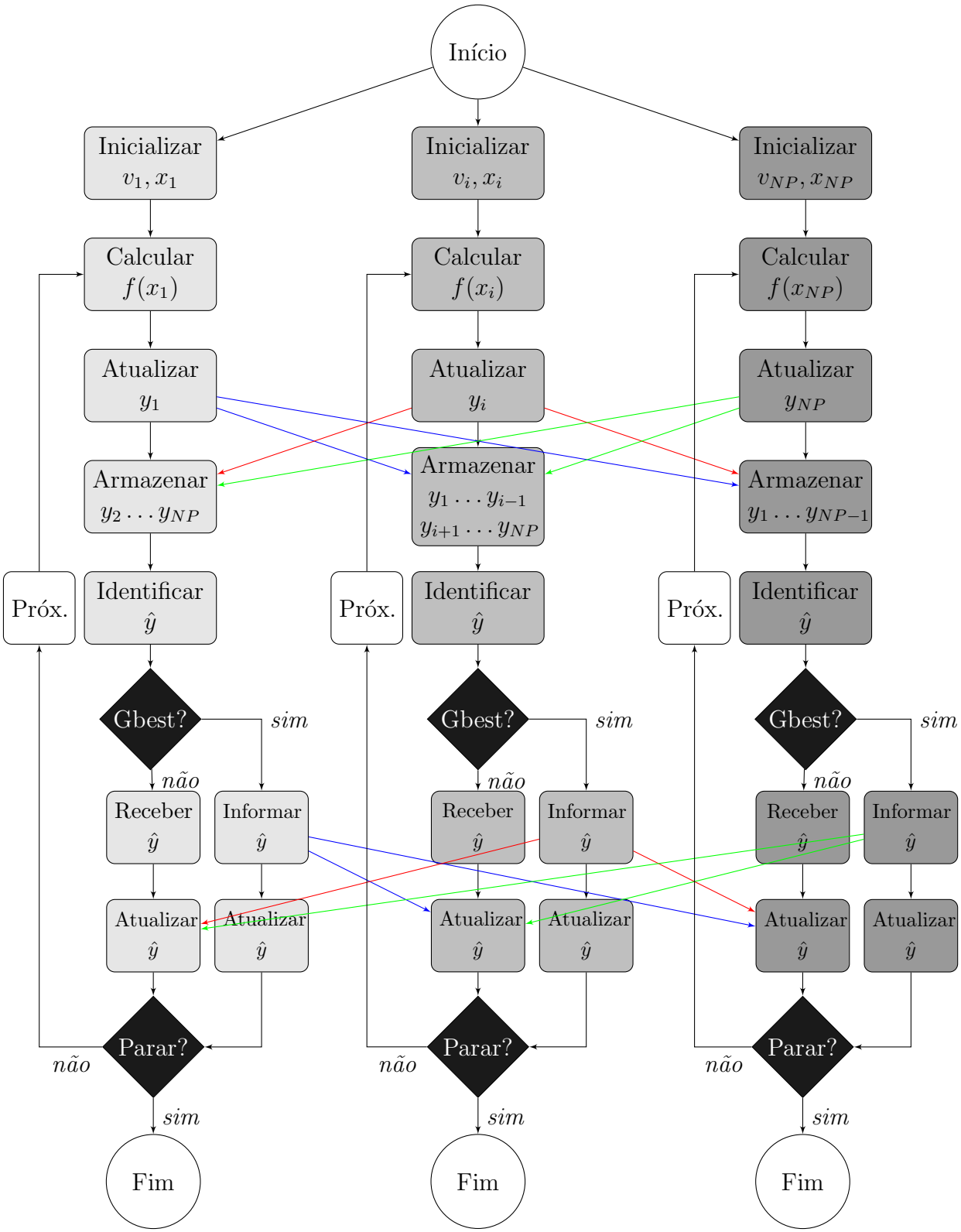


Figura 10: Computação paralela do GBPSO realizada por ADTG para NP partículas

Após a inicialização da velocidade v_i e posição x_i da partícula i, é realizado o cálculo da aptidão através da função objetivo f que elege *Pbest*. Em seguida é realizada a troca de mensagem entre as partículas com o objetivo de informar o valor de *Pbest* as demais partículas. Os valores recebidos são armazenados e comparados, sendo identificado o *Gbest*.

Após a eleição da partícula com *Gbest*, cabe a esta informar às demais partículas o valor de *Gbest* através de um processo de comunicação para que o mesmo seja atualizado em todos os processos paralelos. Em seguida, é realizada a verificação do critério de parada de modo que caso o mesmo seja alcançado, o algoritmo é encerrado. Caso contrário, é realizada uma nova iteração do algoritmo.

Em contraste com o PPSO cuja dinâmica é ilustrada na Figura 9, a atualização do valor de *Gbest* e a verificação do critério de parada são realizados por cada processo. Note que o sincronismo dos processos é garantido pela etapa da comunicação que permite trocar informação sobre o *Gbest*.

Um processo do ADTG mo robo i é estruturado em três etapas principais: *inicialização*, *ajuste* e *execução*, conforme mostrado no Algoritmo 5. O algoritmo é executado de forma distribuída em todos os robôs do enxame. Em ADTG, cada robô id_i representa uma partícula i e possui uma solução de alocação $\mathbb{A}_i$ em potencial para o enxame. O número de dimensões deste problema é representada pelo número de robôs ρ, sendo o espaço de busca em cada dimensão delimitado pelo número de tarefas τ.

Para a implementação de ADTG é proposta uma abordagem discreta do espaço de busca. A motivação para esta abordagem é decorrente da escolha da alocação de tarefas do enxame para representar a posição da partícula no espaço de busca. Desta forma, cada robô id_i possui uma posição no espaço de busca definida pela alocação $\mathbb{A}_i = \{a_1, a_2, \ldots, a_\rho\}$. Cada elemento a_i do conjunto $\mathbb{A}_i$ representa a coordenada da partícula referente a dimensão i do problema, podendo possuir valores dentro do espaço discreto de busca definido por $\mathbb{T} = \{t_1, t_2, \ldots, t_\tau\}$. Vale lembrar que a proporção desejada é representada por $\mathbb{P}$ e que t_i é o identificador da tarefa atualmente alocada ao robô i.

O Algoritmo 5, começa inicializando (linha 1) os parâmetros do problema, conforme será detalhado na Seção 4.3. A cada iteração é verificado se a proporção desejada $\mathbb{P}$ é alcançada (linha 2), comparando-a com a proporção $\mathbb{P}_i$, que é obtida a partir da

Algoritmo 5 ADTG no robô i

Entrada τ, ρ e $\mathbb{P}$;
1: Inicializar(τ, ρ);
2: **Enquanto** *verdadeiro* **Faça**
3: **Se** $\mathbb{P}_i \neq \mathbb{P}$ **Então**
4: AjustarAlocação($\mathbb{A}_i, \mathbb{A}_{Pbest_i}, \mathbb{A}_{Gbest}$);
5: **Senão**
6: ExecutarTarefa(t_i) por um determinado período de tempo;
7: **Fim Se**
8: **Fim Enquanto**

alocação $\mathbb{A}_i$, conforme explicado no Capítulo 1. Caso a proporção desejada não tenha sido alcançada, é realizado o ajuste da alocação $\mathbb{A}_i$ (linha 4). O procedimento usado para ajustar a alocação do robô é inspirado pelo GBPSO, e será detalhado na Seção 4.4. Caso a proporção desejada tenha sido alcançada, a tarefa t_i é realizada por um período de tempo pré-definido (linha 6).

A etapa de execução, implementada por $ExecutarTarefa$, realiza a tarefa t_i (linha 6) alocada para o robô conforme, prescrito na alocação $\mathbb{A}_i$. Em geral, as τ tarefas consideradas podem ter características distintas em relação à ação que deve ser executada e ao tempo de execução desta ação, sendo estas características definidas caso a caso dependendo do problema.

A dinâmica imposta no Algoritmo 5 permite que o robô repita a etapa de ajuste várias vezes, objetivando alcançar uma distribuição das tarefas de acordo com a proporção $\mathbb{P}$ imposta inicialmente. Com essa distribuição de tarefas, os robôs do enxame passam a realizar a tarefa correspondente até que uma nova configuração do problema é informada. Da mesma forma que no caso do ADTL, a configuração do problema é definida pelo número total de robôs ρ, o número total de diferentes tarefas a serem realizadas τ e a proporção desejada $\mathbb{P}$.

4.3 Etapa de Inicialização

A etapa de inicialização é implementada conforme mostrado no Algoritmo 6. A configuração inicial da alocação do robô reflete a diversidade inicial do enxame, que impacta diretamente na eficiência do algoritmo ADTG.

É considerada uma configuração inicial aleatória da alocação do robô, sendo os

Algoritmo 6 Inicializar no robô i

Entrada τ e ρ;
Saída $\mathbb{A}_i$, $\mathbb{A}_{Pbest_i}$, $\mathbb{A}_{Gbest}$, $\mathbb{P}_i$ e t_i;
1: **Para** $r := 1 \rightarrow \rho$ **Faça**
2: $\quad \mathbb{A}_i[r] := GeradorRand(1, \ldots, \tau)$;
3: $\quad \mathbb{C}_i[\mathbb{A}_i[r]] := \mathbb{C}_i[\mathbb{A}_i[r]] + 1$;
4: **Fim Para**
5: **Para** $t := 1 \rightarrow \tau$ **Faça**
6: $\quad \mathbb{P}_i[t] := \mathbb{C}_i[t]/\rho$;
7: **Fim Para**
8: $\mathbb{A}_{Pbest_i} := \mathbb{A}_i$;
9: $\mathbb{A}_{Gbest} := \mathbb{A}_i$;
10: $t_i := \mathbb{A}_{Gbest}[id_i]$;

valores atribuídos às tarefas compreendem valores aleatórios no intervalo definido por $[1, \ldots, \tau]$. Note que nessa codificação t_i é associado com o identificador i. Dessa forma, a alocação $\mathbb{A}_i$ é inicializada com tarefas pertencentes ao conjunto $\mathbb{T}$.

Em seguida, o robô atualiza $\mathbb{A}_i$ com os valores fornecidos pelo gerador de números aleatórios e contabiliza o número de robôs alocados a cada uma das tarefas em τ contadores (linhas 1 – 4). O conjunto dos contadores é representado por $\mathbb{C}_i = \{c_1, c_2, \ldots, c_\tau\}$, onde c_i representa a quantidade de robôs atualmente associados à tarefa de identificador t_i. Em sequência, é atualizado $\mathbb{P}_i$ usando os valores das proporções equivalentes a cada uma das tarefa em relação ao número de robôs ρ (linhas 5 – 7), conforme definido na Equação 1 e na Equação 2 do Capítulo 1. Então, o robô atualiza a sua alocação $\mathbb{A}_{Pbest_i}$ (linha 8) e alocação $\mathbb{A}_{Gbest}$ (linha 9) com $\mathbb{A}_i$, já que se trata do início do processo de otimização, não havendo nenhum passado, e a sua tarefa atual com a tarefa correspondente na alocação $\mathbb{A}_{Gbest}$ (linha 10).

4.4 Etapa de Ajuste

O Algoritmo 7 descreve o método proposto para implementar o ajuste da alocação atual dos robôs do enxame de modo que seja obtida uma nova alocação para os robôs que se aproxime mais da proporção desejada $\mathbb{P}$. A etapa de ajuste é estruturada em 5 sub-etapas:

1. *AtualizarAlocPbest*: permite atualizar a melhor alocação do robô,

2. *IdentificarAlocGbest*: identifica o robô id_{Gbest} com a melhor alocação do enxame,

3. *InformarAlocGbest*: permite ao robô id_{Gbest} informar sua alocação $\mathbb{A}_{Gbest}$ aos demais,

4. *AtualizarAlocGbest*: permite aos demais robôs receber e atualizar $\mathbb{A}_{Gbest}$,

5. *AtualizarAlocAtual*: atualiza a alocação atual.

A função f denota a função objetivo a ser minimizada, e será definida na Seção 4.4.1.

Algoritmo 7 AjustarAlocação no robô i

Entrada $\mathbb{A}_i$, $\mathbb{A}_{Pbest_i}$ e $\mathbb{A}_{Gbest}$;
Saída t_i;
1: AtualizarAlocPbest($\mathbb{A}_i$, $\mathbb{A}_{Pbest_i}$);
2: IdentificarAlocGbest($Pbests$, $\mathbb{A}_{Pbest_i}$);
3: **Se** $id_{Gbest} = id_i$ **Então**
4: InformarAlocGbest($\mathbb{A}_i$);
5: **Senão**
6: AtualizarAlocGbest(id_{Gbest});
7: **Fim Se**
8: **Se** $f(\mathbb{A}_{Gbest}) \neq 0$ **Então**
9: AtualizarAlocAtual($\mathbb{A}_i$, $\mathbb{A}_{Pbest_i}$, $\mathbb{A}_{Gbest}$);
10: **Fim Se**
11: $t_i := \mathbb{A}_{Gbest}[id_i]$;

Inicialmente, é realizada a atualização da melhor alocação do robô i, denominada como $\mathbb{A}_{Pbest_i}$, através de *AtualizarAlocPbest* (linha 1). O procedimento usado para isto é detalhado na Seção 4.4.2. Em seguida, o robô identifica a melhor alocação entre todos os robôs do enxame, denominada como $\mathbb{A}_{Gbest}$, e o identificador id_{Gbest} do robô que possui sua alocação atual $\mathbb{A}_i = \mathbb{A}_{Gbest}$ através de $IdentificarAlocGbest$ (linha 2). Este processo de identificação é explicado na Seção 4.4.3.

A partir deste ponto, o robô id_{Gbest} passa a executar um procedimento distinto daquele realizado pelos demais robôs do enxame. Para isto, os identificadores id_{Gbest} e id_i são comparados (linha 3). O robô de identificação id_{Gbest} informa sua alocação $\mathbb{A}_{Gbest}$ aos demais robôs do enxame através de $InformarAlocGbest$ (linha 4). Este procedimento de comunicação é explicado na Seção 4.4.4. Todo robô cuja identificação é diferente de id_{Gbest} atualiza $\mathbb{A}_{Gbest}$ com a alocação recebida do robô id_{Gbest} através de *AtualizarAlocGbest* (linha 6). Este procedimento é detalhado na Seção 4.4.5.

Em seguida, todos os robôs avaliam a necessidade de atualização da alocação atual $\mathbb{A}_i$, verificando o valor de aptidão $f(\mathbb{A}_{Gbest})$ (linha 8). Um valor de aptidão nulo para $\mathbb{A}_{Gbest}$

identifica a situação onde pelo menos um dos robôs possui uma alocação de tarefas que representa a proporção desejada $\mathbb{P}$. Dessa forma, $f(\mathbb{A}_{Gbest}) = 0$ coincide com o critério de parada $\mathbb{P} = \mathbb{P}_i$ do algoritmo ADTG. Caso o critério de parada não seja alcançado, é realizada a atualização da alocação atual $\mathbb{A}_i$ através de *AtualizarAlocAtual* (linha 9). O procedimento de atualização de $\mathbb{A}_i$ é detalhado na Seção 4.4.6. Ao final, a tarefa atual t_i do robô id_i é atualizada a partir de $\mathbb{A}_{Gbest}$ (linha 11).

4.4.1 Função objetivo para ADTG

A avaliação da aptidão de uma alocação $\mathbb{A}$ é realizada utilizando uma função objetivo $f(\mathbb{A})$, definida pela Equação 11:

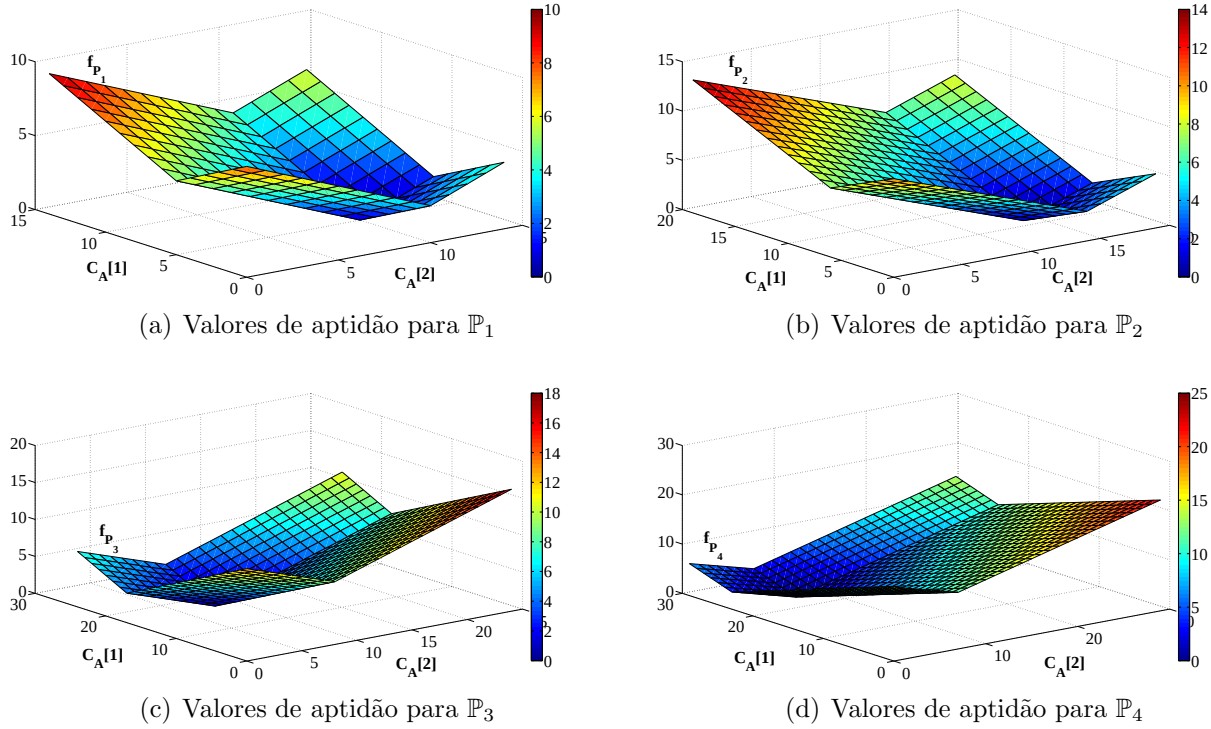
$$f(\mathbb{A}) = \frac{\sum_{i=1}^{\tau} |\mathbb{C}[i] - \mathbb{C}_{\mathbb{A}}[i]|}{\tau}, \tag{11}$$

onde $\mathbb{C}$ representa a quantidade desejada de robôs alocados a cada tarefa de acordo com a proporção desejada $\mathbb{P}$ e $\mathbb{C}_{\mathbb{A}}$ representa quantidade de robôs alocados a cada tarefa de acordo com a alocação $\mathbb{A}$.

Na Figura 11 são mostrados os gráficos dos possíveis valores de aptidão $f(\mathbb{A})$ para quatro exemplos de proporções. Cada exemplo representa um problema de alocação, onde um conjunto de duas tarefas $\mathbb{T} = \{t_1, t_2\}$ ($\tau = 2$) deve ser alocado à um enxame de robôs. Assim, os gráficos $f_{P_1}(\mathbb{A})$, $f_{P_2}(\mathbb{A})$, $f_{P_3}(\mathbb{A})$ e $f_{P_4}(\mathbb{A})$ correspondem as proporções objetivo, $\mathbb{P}_1 = \{\frac{5}{15}, \frac{10}{15}\}$, $\mathbb{P}_2 = \{\frac{6}{20}, \frac{14}{20}\}$, $\mathbb{P}_3 = \{\frac{17}{25}, \frac{8}{25}\}$ e $\mathbb{P}_4 = \{\frac{23}{30}, \frac{7}{30}\}$. Essas proporções são associadas as quantidades de robôs alocados a cada tarefa $\mathbb{C}_1 = \{5, 10\}$, $\mathbb{C}_2 = \{6, 14\}$, $\mathbb{C}_3 = \{17, 8\}$ e $\mathbb{C}_4 = \{23, 7\}$, respectivamente.

Nos gráficos, a função objetivo $f(\mathbb{A})$ apresenta um valor nulo quando a alocação $\mathbb{A}$ satisfaz a proporção desejada $\mathbb{P}$ para o problema de alocação. Dessa forma, as quatro funções f_{P_1}, f_{P_2}, f_{P_3} e f_{P_4} convergem para seus respectivos mínimos para todas as alocações, $\mathbb{A}_1$, $\mathbb{A}_2$, $\mathbb{A}_3$ e $\mathbb{A}_4$, que apresentam uma distribuição das tarefas, tais que $\mathbb{C}_{\mathbb{A}_1} = \{5, 15\}$, $\mathbb{C}_{\mathbb{A}_2} = \{6, 14\}$, $\mathbb{C}_{\mathbb{A}_3} = \{17, 8\}$ e $\mathbb{C}_{\mathbb{A}_4} = \{23, 7\}$, respectivamente.

Note que a função objetivo avalia a aptidão da partícula de forma indireta, uma vez que é baseada nos números totais de robôs alocados a cada uma das tarefas. Dessa forma, alocações $\mathbb{A}$ que são permutações tem a mesma quantidade de robôs alocados a cada tarefa ($\mathbb{C}_{\mathbb{A}}$) e em consequência o mesmo valor de aptidão $f(\mathbb{A})$.



(a) Valores de aptidão para $\mathbb{P}_1$

(b) Valores de aptidão para $\mathbb{P}_2$

(c) Valores de aptidão para $\mathbb{P}_3$

(d) Valores de aptidão para $\mathbb{P}_4$

Figura 11: Representação gráfica de f_{P_1}, f_{P_2}, f_{P_3} e f_{P_4}

Como exemplo ilustrativo, seja um enxame de 8 robôs com duas tarefas a serem alocadas respeitando uma proporção $\mathbb{P}$. Considere o grupo de 3 alocações possíveis definidas por $A_1 = \{2, 2, 3, 1, 0, 2, 0, 1\}$, $A_2 = \{3, 2, 0, 1, 1, 2, 2, 0\}$ e $A_3 = \{1, 0, 1, 3, 2, 0, 2, 2\}$, onde os identificadores das tarefas são representados pelo conjunto $\mathbb{T} = \{0, 1, 2, 3\}$. Desta forma, cada uma das três alocações possui uma mesma quantidade de robôs alocados as tarefas definida por $\mathbb{C}_{\mathbb{A}_1} = \mathbb{C}_{\mathbb{A}_2} = \mathbb{C}_{\mathbb{A}_3} = \{2, 2, 3, 1\}$, ou seja, dois robôs alocados à tarefa 0, dois robôs alocados à tarefa 1, três robôs alocados à tarefa 2 e um robô alocado à tarefa 3. Assim, as três alocações possuirão o mesmo valor de aptidão.

4.4.2 Atualização da Melhor Alocação Local

O Algoritmo 8 descreve o processo de atualização da melhor alocação $\mathbb{A}_{Pbest_i}$ alcançada no passado pela partícula representada pelo robô i. Com este objetivo, é realizada a comparação entre o valor de aptidão da alocação atual $\mathbb{A}_i$ e o valor de aptidão da alocação $\mathbb{A}_{Pbest_i}$ obtida até o momento (linha 1). Caso $f(\mathbb{A}_i)$ apresente um valor menor ou igual que $f(\mathbb{A}_{Pbest_i})$, a melhor alocação atual $\mathbb{A}_{Pbest_i}$ será atualizada com $\mathbb{A}_i$ (linha 2).

Em seguida, é realizado um processo de comunicação para troca de informação

Algoritmo 8 AtualizarAlocPbest no robô i

Entrada ρ, $\mathbb{A}_i$ e $\mathbb{A}_{Pbest_i}$;
Saída $Pbests$ e $\mathbb{A}_{Pbest_i}$;
1: **Se** $f(\mathbb{A}_i) \leq f(\mathbb{A}_{Pbest_i})$ **Então**
2: $\quad \mathbb{A}_{Pbest_i} := \mathbb{A}_i$;
3: **Fim Se**
4: $msg \leftarrow \langle id_i, f(A_{Pbest_i})\rangle$;
5: **Repita**
6: $\quad$ **Enviar** msg aos demais $\rho - 1$ robôs;
7: $\quad$ **Receber** msg dos demais $\rho - 1$ robôs;
8: **Até** receber de todos os robôs
9: **Para** $r := 1 \rightarrow \rho - 1$ **Faça**
10: $\quad$ **Se** $\mathbb{I}[r] \neq id_i$ **Então**
11: $\quad\quad Pbests[r] := f(\mathbb{A}_{Pbest_r})$;
12: $\quad$ **Fim Se**
13: **Fim Para**

entre os robôs (linhas 4 – 6). Esta troca de mensagens tem como finalidade informar os demais robôs o valor de aptidão da melhor alocação $f(\mathbb{A}_{Pbest_i})$ de cada robô id_i e receber a mesma informação dos demais, de modo que ao término do processo de comunicação, os robôs do enxame possuam pleno conhecimento da melhor alocação do enxame. Para isto, o robô i envia uma mensagem composta por id_i e $f(\mathbb{A}_{Pbest_i})$ aos demais $\rho - 1$ robôs do enxame. De forma complementar, são recebidas mensagens com conteúdo análogo dos demais robôs. Ao completar este processo, o robô atualiza o vetor $Pbests$ com os valores de aptidão recebidos pelas mensagens dos demais robôs (linhas 10 – 12).

4.4.3 Identificação do Robô Gbest

A identificação do robô que está com a melhor alocação global entre todos os robôs do enxame, denominado id_{Gbest}, é descrita no Algoritmo 9. Este algoritmo identifica o menor valor em $Pbests$ assim como a identificação do robô associado. No algoritmo, são definidas as variáveis $Pbest_{min}$ e id', utilizadas para armazenar, respectivamente, o menor valor de aptidão local e o identificador do robô que possui esta menor aptidão. Inicialmente, id' recebe o identificador id_i do robô (linha 1) enquanto que $Pbest_{min}$ recebe o valor de aptidão $f(\mathbb{A}_{Pbest_i})$ (linha 2).

A identificação do robô id_{Gbest} é realizada por todos os robôs do enxame, de modo que todos cheguem ao mesmo resultado no final de $\rho - 1$ comparações (linhas 3 – 8). A identificação do robô id_{Gbest} pelos robôs se deve ao fato de que todos os robôs recebem

Algoritmo 9 IdentificarAlocGbest no robô i

Entrada ρ, $\mathbb{A}_{Pbest_i}$ e $Pbests$;
Saída id_{Gbest};
1: $id' := id_i$;
2: $Pbest_{min} := f(\mathbb{A}_{Pbest_i})$;
3: **Para** $r := 1 \rightarrow \rho - 1$ **Faça**
4: **Se** $Pbest_{min} > Pbests[r]$ **Então**
5: $Pbest_{min} := Pbests[r]$;
6: $id' := \mathbb{I}[r]$, onde $\mathbb{I}[r] \neq id_i$;
7: **Fim Se**
8: **Fim Para**
9: $id_{Gbest} := id'$;

$Pbests$.

4.4.4 Envio da Melhor Alocação Global

Nesta etapa, o robô identificado como id_{Gbest} informa aos demais robôs sua alocação atual através de um processo de troca de mensagens, conforme mostra o Algoritmo 10. Ao iniciar o algoritmo, é realizada a atualização da melhor alocação global $\mathbb{A}_{Gbest}$ com a alocação atual $\mathbb{A}_i$ do robô (linha 1).

Em seguida, é realizado o processo de troca de mensagens entre os robôs (linhas 4 – 6). Neste processo de comunicação, o robô id_{Gbest} envia mensagens, denominadas $msg1$, aos demais robôs e recebe dos demais mensagens, denominadas $msg2$. A mensagem $msg1$ é composta por id_{Gbest} e $\mathbb{A}_{Gbest}$, de modo que ao ser enviada, informe aos demais $\rho - 1$ robôs do enxame a alocação $\mathbb{A}_{Gbest}$.

Algoritmo 10 InformarAlocGbest no robô i

Entrada ρ, $\mathbb{A}_i$;
Saída $\mathbb{A}_{Gbest}$;
1: $\mathbb{A}_{Gbest} := \mathbb{A}_i$;
2: $msg1 \leftarrow \langle id_i, \mathbb{A}_{Gbest} \rangle$;
3: **Repita**
4: **Enviar** $msg1$ aos demais $\rho - 1$ robôs;
5: **Receber** $msg2$ dos demais $\rho - 1$ robôs;
6: **Até** que todos tenham recebido

A mensagem $msg2$ é composta pelo identificador do robô emissor. Ao ser recebida pelo robô id_{Gbest}, esta mensagem confirma o recebimento de $msg1$ pelo robô emissor. Dessa

forma, robô id_{Gbest}, ao receber a mensagem $msg2$ de todos os demais robôs, identifica que todos já receberam a alocação $\mathbb{A}_{Gbest}$.

4.4.5 Atualização da Melhor Alocação Global

No Algoritmo 11, o robô de identificador $id_i \neq id_{Gbest}$ é informado sobre a melhor alocação global $\mathbb{A}_{Gbest}$ pertencente ao robô id_{Gbest} através de um processo de troca de mensagens (linhas 1 – 3). Este processo de comunicação é análogo ao processo realizado em $InformarAlocGbest$.

Inicialmente o robô aguarda pelo recebimento de uma mensagem proveniente do robô id_{Gbest} contendo a sua alocação $\mathbb{A}_{id_{Gbest}}$. Ao receber $msg1$, o robô envia uma mensagem de confirmação de recebimento composta pelo seu id_i. Em seguida, o robô atualiza a sua alocação global $\mathbb{A}_{Gbest}$ com a alocação $\mathbb{A}_{id_{Gbest}}$ recebida (linha 4).

Algoritmo 11 AtualizarAlocGbest no robô i

Entrada id_{Gbest};
Saída $\mathbb{A}_{Gbest}$;
1: $ms2g \leftarrow \langle id_i \rangle$;
2: **Receber** $msg1$ do robô id_{Gbest};
3: **Enviar** $msg2$ ao robô id_{Gbest};
4: $\mathbb{A}_{Gbest} := \mathbb{A}_{id_{Gbest}}$;

Note que a execução do Algoritmo 11 pelo robô id_i ocorre ao mesmo tempo que a execução do Algoritmo 10 pelo robô id_{Gbest}. Dessa forma, durante a troca de mensagens entre os robôs, a mensagem $msg1$, proveniente do robô id_{Gbest}, será destinada aos demais robôs do enxame, que confirmarão o seu recebimento enviando a mensagem $msg2$ para o robô id_{Gbest}.

4.4.6 Atualização da Alocação

O Algoritmo 12 realiza a atualização da alocação atual $\mathbb{A}_i$ baseando-se nas contribuições de alocação cognitiva e social, análogas aos componentes cognitivo e social do PSO. As contribuições atualizam a velocidade do robô nas ρ dimensões da posição definida pela alocação.

Para cada dimensão r, é realizada a atualização do valor de $velocidade[r]$ de acordo com a Equação 9, mostrada na Seção 4.1.1 deste capítulo. Inicialmente, é realizado o cálculo dos componentes cognitivo (linha 4) e social (linha 5) da dimensão.

Algoritmo 12 AtualizarAlocAtual no robô i

Entrada τ, ρ, $\mathbb{A}_i$, $\mathbb{A}_{Pbest_i}$, $\mathbb{A}_{Gbest}$;
Saída $\mathbb{A}_i$ e $\mathbb{P}_i$;

1: **Para** $r := 1 \rightarrow \rho$ **Faça**
2: $r_1 := GeradorRand[0 \ldots 1]$;
3: $r_2 := GeradorRand[0 \ldots 1]$;
4: $\mathbb{A}_{cognitiva} := c_1 \times r_1 \times (\mathbb{A}_{Pbest_i}[r] - \mathbb{A}_i[r])$;
5: $\mathbb{A}_{social} := c_2 \times r_2 \times (\mathbb{A}_{Gbest}[r] - \mathbb{A}_i[r]$;
6: $velocidade[r] := velocidade[r] + \mathbb{A}_{cognitiva} + \mathbb{A}_{social}$;
7: **Se** $\{velocidade[r]\} \leq 0.5$ **Então**
8: $velocidade[r] := \lfloor velocidade[r] \rfloor$;
9: **Senão**
10: $velocidade[r] := \lceil velocidade[r] \rceil$;
11: **Fim Se**
12: $\mathbb{A}_i[r] := \mathbb{A}_i[r] + velocidade[r]$;
13: **Se** $\mathbb{A}_i[r] < t_1$ **Então**
14: $\mathbb{A}_i[r] := t_1$;
15: **Senão**
16: **Se** $\mathbb{A}_i[r] > t_\tau$ **Então**
17: $\mathbb{A}_i[r] := t_\tau$;
18: **Fim Se**
19: **Fim Se**
20: $\mathbb{C}_i[\mathbb{A}_i[r]] := \mathbb{C}_i[\mathbb{A}_i[r]] + 1$;
21: **Fim Para**
22: **Para** $t := 1 \rightarrow \tau$ **Faça**
23: $\mathbb{P}_i[t] := \mathbb{C}_i[t]/\rho$;
24: **Fim Para**

Como forma de adequar o resultado da posição ao espaço de busca discreto, é realizado um arredondamento do valor de $velocidade[r]$, de modo que $velocidade[r] \in \mathbb{Z}$ (linhas 7 – 11). O processo de arredondamento de um número $x \mid x \in \mathbb{Q}$ considera a parte fracionária $\{x\}$ para definir se o número terá o seu valor arredondado. Para o processo de arredondamento utilizado considerou-se o valor de referência de $0,5$ para ser comparado a $\{x\}$. Caso $\{x\}$ seja menor ou igual ao valor de referência, x terá seu valor atualizado para o maior valor inteiro anterior a ele, representado por $\lfloor x \rfloor$. Caso contrário, $\{x\}$ terá seu valor atualizado para o maior valor inteiro posterior a ele, representado por $\lceil x \rceil$.

Em seguida, é realizada a atualização de $\mathbb{A}_i[r]$ com o seu valor anterior acrescido de $velocidade[r]$ (linha 12). O valor de $\mathbb{A}_i[r]$ é delimitado dentro do espaço de busca definido pelos valores do conjunto dos identificadores das tarefas $\mathbb{T} = \{t_1, t_2, \ldots, t_\tau\}$ (linhas 13 – 19), de modo que os valores estarão sempre compreendidos entre os extremos inferior

t_1 e superior t_τ. Durante o processo de atualização de $\mathbb{A}_i$, é realizada a contabilização do número de robôs alocados a cada uma das tarefas em τ contadores $\mathbb{C}_i$, baseando na alocação $\mathbb{A}_i$ atualizada (linha 20). Vale lembrar que o conjunto dos contadores é representado por $\mathbb{C}_i$, onde $\mathbb{C}_i = \{c_1, c_2, \ldots, c_\tau\}$ e c_i representa a quantidade de robôs atualmente associados à tarefa de identificador i.

Em seguida, é realizada a atualização de $\mathbb{P}_i$ através do cálculo das proporções equivalentes a cada uma das tarefas em relação ao número de robôs ρ (linhas 22 - 24), conforme definido na Equação 1 e na Equação 2 do Capítulo 1.

4.5 Considerações Finais do Capítulo

Neste capítulo foi apresentado um algoritmo com abordagem global de alocação dinâmica de tarefas ADTG. O algoritmo foi desenvolvido como uma solução estocástica ao problema de alocação de tarefas em grupos de robôs. O mesmo baseia-se na estrutura do algoritmo *Global Best PSO* com o objetivo de minimizar o valor de aptidão das alocações apresentadas como solução pelos robôs do enxame. Dessa forma, a solução que apresentar um valor de aptidão nulo é considerada a nova alocação para o enxame, de modo que cada robô atualiza sua tarefa de acordo com a alocação encontrada fazendo com que o enxame alcance a convergência para a proporção desejada. O capítulo seguinte apresenta uma descrição das implementações realizadas para os dois algoritmos propostos.

Capítulo 5

IMPLEMENTAÇÃO DOS ALGORITMOS PROPOSTOS

NESTE capítulo é detalhada a implementação dos algoritmos ADTL e ADTG utilizando robôs reais, descritos no Capítulo 3 e Capítulo 4, respectivamente.

Na Seção 5.1 são apresentados os detalhes do enxame e do robô utilizado na implementação, juntamente com os detalhes da comunicação base entre os robôs do enxame. A Seção 5.2 e Seção 5.3 detalham a composição das mensagens utilizadas pelo processo de comunicação realizado na implementação dos algoritmos ADTL e ADTG, respectivamente. A Seção 5.4 apresenta as considerações finais para o capítulo.

5.1 O Robô

Para a implementação dos algoritmos propostos foram utilizados enxames de robôs do tipo Elisa III, mostrados na Figura 12. A comunicação entre os robôs é realizada de modo *wireless*, de modo que não impeça a movimentação dos robôs. Na comunicação é realizada a troca de mensagens entre os robôs, sendo cada mensagem formada por uma estrutura definida pelo protocolo de comunicação. As mensagens geradas durante a comunicação são identificadas com endereços de origem ou destino, de acordo com a necessidade da comunicação.

O número de robôs ρ no enxame possui um impacto direto no processo de comunicação, visto que acarreta um aumento no fluxo de mensagens trocadas, proporcional ao aumento de robôs no enxame. Com o aumento do fluxo de mensagens o gerenciamento do processo de entrega e recebimento de mensagens, realizado pelo dispositivo de comu-

(a) Detalhe de um robô no enxame (b) Enxame composto por 24 robôs

Figura 12: Enxames de Robôs Elisa III

nicação embarcado no robô, é degradado de modo que mensagens que não possam ser armazenadas no *buffer* do dispositivo sejam perdidas.

Para suprir o aumento no fluxo de mensagens, é proposto um protocolo de comunicação otimizado que possibilita aumentar a velocidade de comunicação para a troca de mensagem entre os robôs. Com o aumento na velocidade de comunicação, as mensagens armazenadas nos *buffers* de entrada e saída do dispositivo de comunicação são tratadas de forma mais rápida diminuindo a incidência de perda de mensagens.

5.1.1 Arquitetura do Robô

O robô Elisa III, mostrado na Figura 13, é um robô móvel que possui uma série de dispositivos embarcados em uma estrutura de pequena dimensão, com aproximadamente 5 e 3 centímetros de diâmetro e altura, respectivamente. O funcionamento dos dispositivos é gerenciado por um microcontrolador Atmel ATmega2560 de 8 MHz (NORDIC, 2008) aliado a uma memória RAM de 8 KB. O robô possui outros dois tipos de memória embarcadas, sendo uma memória *Flash* com capacidade de 256 KB para o armazenamento do programa à ser executado e uma memória EEPROM com capacidade de 4 KB responsável por armazenar instruções específicas ao funcionamento do robô. O *id* do robô é armazenado nesta memória que pode ser acessada a qualquer momento da execução do programa embarcado.

O robô Elisa III é equipado com um LED RGB central capaz de produzir qualquer cor através da combinação de intensidade das cores vermelho, verde e azul. A cor obtida é intensificada por um difusor localizado na parte superior do robô.

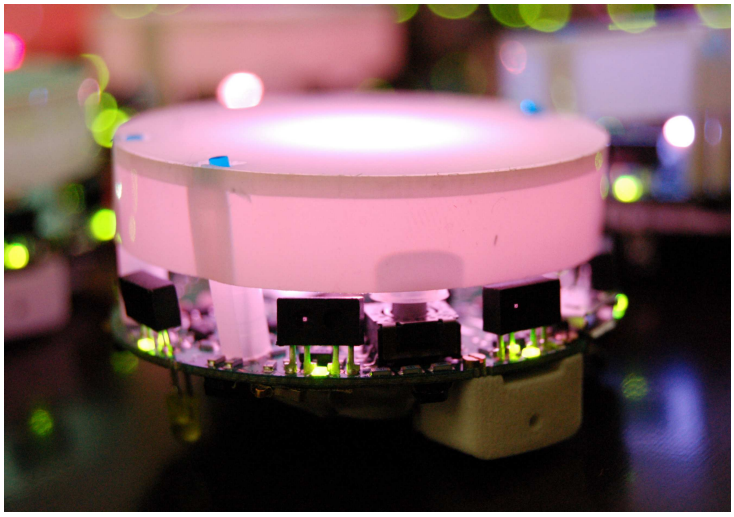


Figura 13: Robô Elisa III

A plataforma de compilação AVR-GCC do programa à ser executado pelo robô permite a programação nas linguagens C e C++, sendo compatível com a plataforma Arduino IDE (IDE, 2014).

Cada robô é equipado com um módulo de comunicação transmissor/receptor adequado para aplicações de comunicação *wireless* com consumo de energia ultra-baixo, e que opera em Radio Frequência (RF). O módulo de comunicação é designado para operar na faixa de rádio ISM (*Industrial, Scientific and Medical band basis*) na frequência de operação de 2.4 GHz.

5.1.2 Comunicação por Radio Frequência

O processo de comunicação caracteriza-se pela troca de mensagens entre os robôs do enxame com o objetivo de realizar uma troca de informações necessárias aos algoritmos de alocação de tarefas propostos. Dessa forma, cada robô comunica-se com os demais robôs por RF utilizando o módulo de comunicação embarcado em seu hardware por intermédio de uma estação base conectada a um computador, conforme mostra a Figura 14.

A estação base é conectada a um computador via conexão USB e realiza a transferência das mensagens de e para os robôs de modo wireless. Assim, o módulo de comunicação do robô comunica-se com o microcontrolador via comunicação SPI (Serial Peripheral Interface) recebendo os dados à serem enviados aos demais robôs. Em seguida, o módulo de comunicação realiza o encapsulamento dos dados a serem enviados em uma estrutura contendo as informações necessárias para o envio, formando as mensagens que são envia-

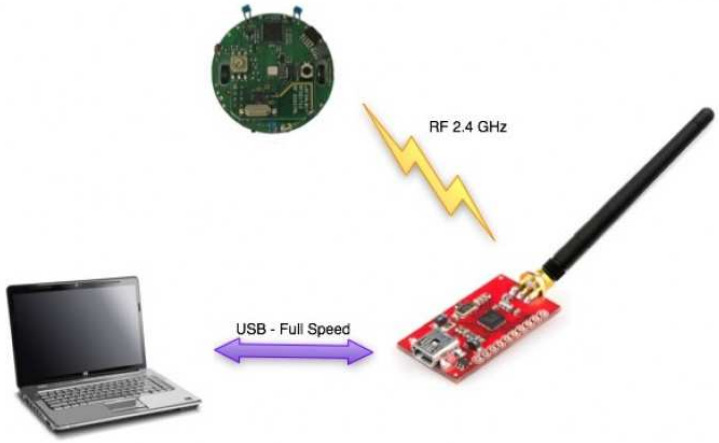


Figura 14: Comunicação de e para o robô via RF

das para a estação base. Ao receber as mensagens, a estação base interpreta-as e as envia para o seu endereço de destino. As mensagens provenientes da estação base são recebidas pelos robôs via o módulo de comunicação, que interpreta-as e informa ao microcontrolador os dados contidos nas mensagens.

Cada robô é identificado por um único identificador *id* no enxame que representa o seu endereço no processo de comunicação. Este identificador é armazenado em um endereço de memória específico na memória EEPROM do robô. Toda mensagem proveniente da estação base tem um endereço de destino que deve coincidir com um dos endereços dos robôs no enxame.

Uma vez recebida a mensagem, o robô compara o endereço de destino da mensagem recebida com o seu endereço para avaliar se ele é o destinatário desta mensagem. Caso a mensagem seja destinada a ele, a mesma é armazenada e interpretada pelo robô.

A comunicação entre os robôs é intermediada pela estação base, que é gerenciada continuamente pelo computador. As mensagens recebidas pela estação base são armazenadas em um buffer na própria estação base e posteriormente direcionadas ao computador que as interpreta e realiza o processo de encapsulamento com o endereço de destino. O computador verifica a existência de novas mensagens no buffer da estação base uma vez a cada milissegundo, sendo esta a restrição para a máxima velocidade da comunicação. Como forma de compensar esta limitação, é implementado um protocolo de comunicação otimizado onde os pacotes que trafegam entre o computador e a estação base contem mensagens para quatro robôs simultaneamente. A estação base é responsável por separar

o pacote recebido em quatro pacotes individuais de 16 bytes cada, antes de formar as quatro mensagens e envia-las para o endereço de destino indicado.

O mesmo procedimento é realizado durante a recepção das mensagens provenientes do robôs. Neste caso, a estação base é responsável por receber as mensagens de quatro robôs e formar um único pacote de 64 bytes que será enviado ao computador. A implementação deste protocolo de comunicação otimizado melhora o rendimento da comunicação, tornando-a quatro vezes mais rápida.

Na Figura 15 é mostrada a estrutura da mensagem de 16 bytes enviada pelo robô para a estação base. O primeiro byte é definido pelo *byte de validade* (BV), responsável por validar a autenticidade da mensagem enviada pelo robô quando a mesma é recebida na estação base. Os dois bytes seguintes, contem o *id* do robô emissor da mensagem e o último byte define o tipo da mensagem, no caso de existirem dois ou mais tipos diferentes de mensagens no processo de comunicação. Os 12 bytes restantes contem a *carga útil* da mensagem, que são os dados à serem enviados.

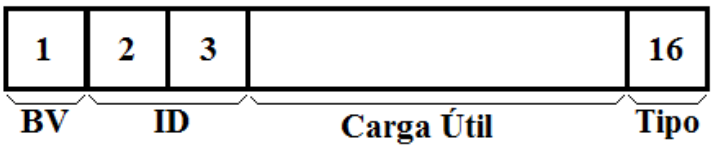


Figura 15: Estrutura da mensagem enviada pelo robô para a estação base

A estrutura da mensagem enviada pela estação base para o robo *id*, mostrada na Figura 16, é semelhante a estrutura da mensagem mostrada na Figura 15, sendo igualmente composta por 16 bytes. O primeiro byte é definido pelo *byte de validade* (BV), que valida a autenticidade da mensagem enviada pela estação base quando a mesma é recebida pelo robô. O byte de número 14 define o tipo da mensagem, quando existirem dois ou mais tipos de mensagens, e os dois últimos bytes contem o endereço de destino do robô *id*. Os 12 bytes restantes contem a *carga útil* da mensagem à ser recebida pelo robô.

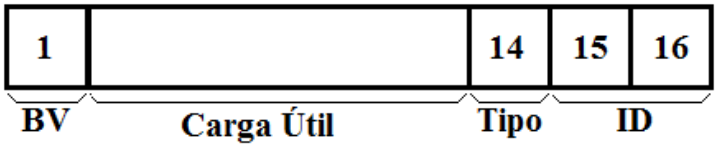


Figura 16: Estrutura da mensagem enviada pela estação base para o robô *id*

5.2 Implementação em ADTL

O processo de comunicação implementado no algoritmo ADTL é utilizado na Etapa de Atualização, apresentada na Seção 3.2 do Capítulo 3. Nesta implementação, é necessário um único tipo de mensagem no processo de comunicação, de modo que o byte correspondente ao tipo da mensagem não é utilizado.

Neste algoritmo, a carga útil da mensagem gerada pelo robô id_i e enviada para a estação base é formada unicamente por um byte que corresponde a tarefa t_i atualmente alocada à este robô. Para compor a informação necessária aos demais robôs são utilizados também os bytes que contem a informação id_i do robô que enviou a mensagem. Assim, a informação a ser repassada aos demais robôs do enxame é composta por três bytes ao todo, sendo dois bytes para id_i e um byte para t_i.

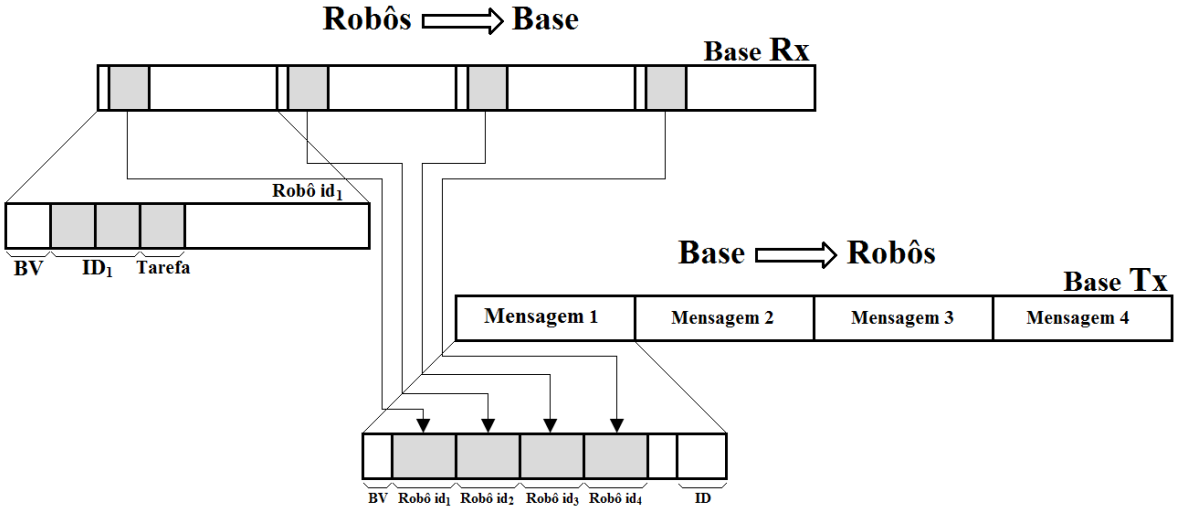


Figura 17: Formato do pacote das mensagens no processo de comunicação em ADTL

A formação dos pacotes durante o processo de comunicação entre os robôs para o algoritmo ADTL é mostrado na Figura 17. No momento em que o computador verifica a existência de novas mensagens no buffer da estação base, é formado o pacote de 64 bytes contendo quatro mensagens cada. O pacote formado é enviado ao computador que interpreta o conteúdo das mensagens identificando a carga útil de cada uma das mensagens que o compõem. Em seguida, a identificação da tarefa t_i presente na carga útil de cada uma das quatro mensagens recebidas e os dois bytes correspondentes ao id_i do robô são copiados para formar uma nova mensagem à ser enviada. Dessa forma, a nova

mensagem gerada contem como carga útil a informação contida nas quatro mensagens recebidas pela estação base. O conteúdo da mensagem gerada é replicado em quatro novas mensagens, sendo cada mensagem destinada a um endereço de destino *id* diferente. As quatro mensagens geradas formam o pacote para ser encaminhado novamente à estação base pelo computador. Ao receber o pacote gerado pelo computador, a estação base separa o pacote em quatro mensagens e as envia aos robôs cujo identificadores id_i estão mencionados no endereço de destino *id* de cada mensagem.

5.3 Implementação em ADTG

Para implementar o processo de comunicação no algoritmo ADTG são considerados dois momentos onde a troca de mensagens entre os robôs é necessária. O primeiro processo de comunicação consiste na troca de mensagens entre todos os robôs do enxame com o objetivo de informar o valor de aptidão $f(\mathbb{A}_{Pbest})$ da melhor alocação local aos demais robôs. Este processo é realizado na Seção 4.4.2 do Capítulo 4.

O segundo processo de comunicação é realizado através da troca de mensagens entre o robô id_{Gbest} e os demais robôs do enxame. Possui o objetivo de informar e atualizar a melhor alocação global $\mathbb{A}_{Gbest}$ do enxame, conforme detalha a Seção 4.4.4 e Seção 4.4.5 do Capítulo 4, respectivamente. Dessa forma, são gerados dois tipos diferentes de mensagens neste processo de comunicação. Sendo um tipo de mensagem gerada por id_{Gbest} para ser enviada aos demais robôs informando $\mathbb{A}_{Gbest}$, e um outro tipo gerado pelos demais robôs para ser enviada ao robô id_{Gbest} como forma de confirmação do recebimento de sua mensagem.

Desta forma, teremos ao todo três tipos diferentes de mensagens para compor a comunicação realizada em ADTG. Na Figura 18 são mostrados os três tipos de mensagens formados para o algoritmo ADTG.

A mensagem de Tipo 1 corresponde ao primeiro processo de comunicação, utilizando os dois bytes que contem a informação id_i e o byte referente a $f(\mathbb{A}_{Pbest})$ da carga útil para compor a informação necessária à ser enviada aos demais robôs. As mensagens de Tipos 2 e 3 correspondem ao segundo processo de comunicação. No Tipo 2 são utilizados os 10 primeiros bytes da carga útil, que refere-se a $\mathbb{A}_{Gbest}$, como informação a ser enviada aos demais robôs do enxame, enquanto que na mensagem de Tipo 3 são utilizados

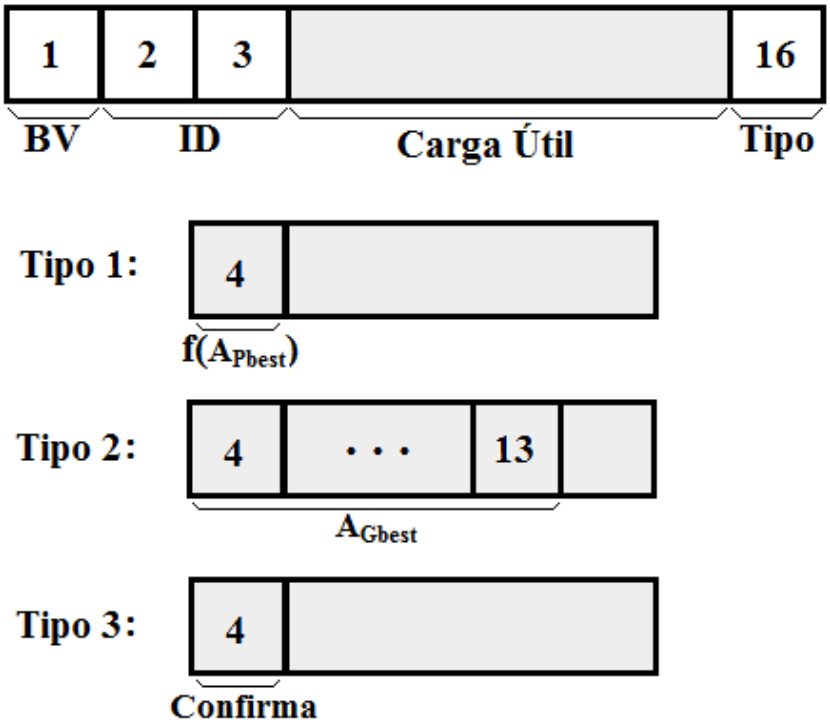


Figura 18: Tipos de mensagens enviadas pelo robô em ADTG

os bytes referentes à id_i e o byte da carga útil referente a *Confirmação* para compor a informação.

Para o primeiro processo de comunicação (Tipo 1), o processo de formação dos pacotes durante a troca de mensagens entre os robôs é semelhante ao implementado em ADTL, conforme mostra a Figura 19. Entretanto, para a implementação em ADTG é utilizado o byte referente ao Tipo da mensagem que conterá o valor 1, indicando que as mensagens geradas são do primeiro tipo.

No segundo processo de comunicação (Tipos 2 e 3), a troca de mensagens é realizada entre dois grupos de robôs, sendo um grupo formado pelo robô id_{Gbest} e o outro grupo formado pelos demais robôs do enxame. O grupo formado unicamente pelo robô id_{Gbest} enviará mensagens do Tipo 2 aos demais robôs do outro grupo informando a sua alocação $\mathbb{A}_{Gbest}$, enquanto que os demais robôs, em resposta ao recebimento desta mensagem, enviarão mensagens do Tipo 3 unicamente para o robô id_{Gbest}, confirmando o seu recebimento.

Dessa forma, existem nesta comunicação dois tipos de mensagem sendo trocadas entre os robôs simultaneamente, de modo que os pacotes no segundo processo de co-

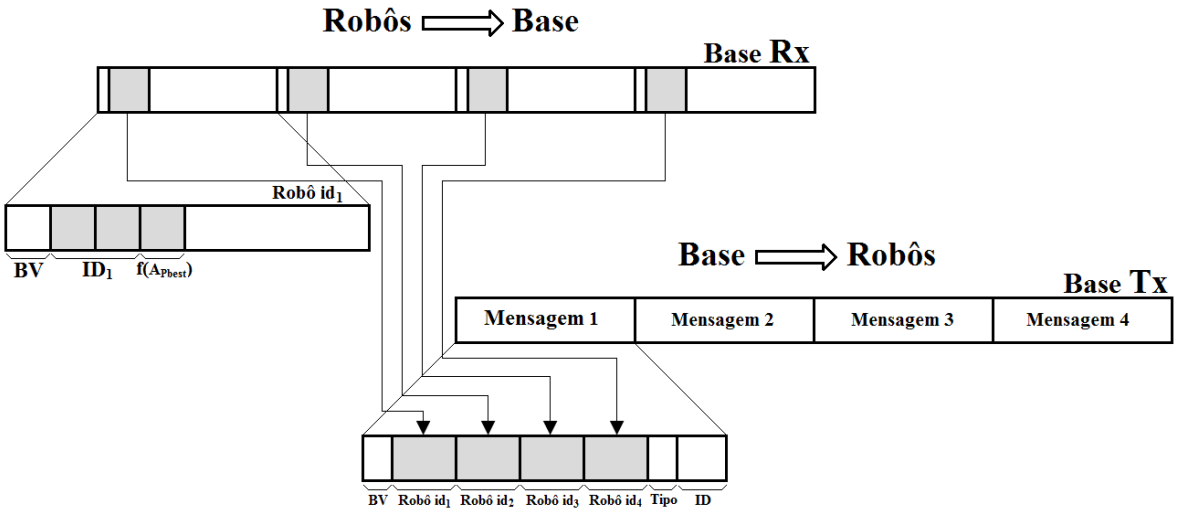


Figura 19: Formato do pacote das mensagens de Tipo 1 para o primeiro processo de comunicação em ADTG

municação são formados pela composição de dois tipos de mensagens. A formação dos pacotes é mostrada na Figura 20 através de um exemplo ilustrativo, onde as três primeiras mensagens do pacote são do Tipo 3 e a quarta é uma mensagem do Tipo 2.

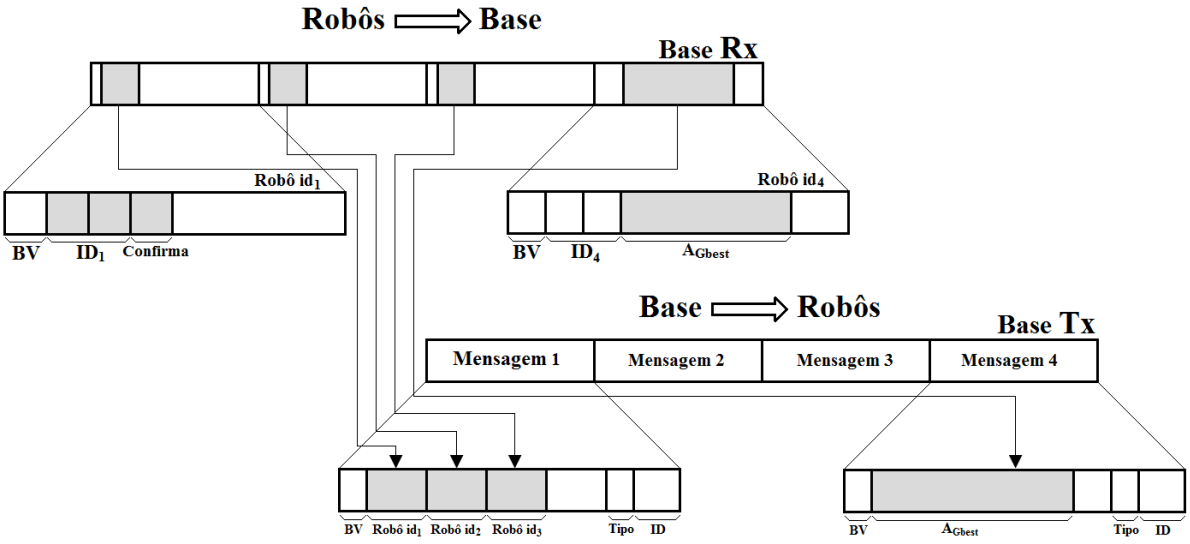


Figura 20: Formato do pacote das mensagens de Tipos 2 e 3 para o segundo processo de comunicação em ADTG

Para a formação das mensagens do Tipo 2, geradas pelo robô id_{Gbest}, é realizada uma codificação da alocação $\mathbb{A}_{Gbest}$. A codificação consiste em converter a alocação de tarefas em 10 estruturas binárias de 8 bits cada, para que possam compor os 10 bytes disponibilizados para a alocação na carga útil da mensagem. Na decodificação, realizada

pelos demais robôs do enxame ao receberem a mensagem de Tipo 2, é realizado o processo inverso com o objetivo de recompor a alocação $\mathbb{A}_{Gbest}$.

No processo de codificação de $\mathbb{A}_{Gbest}$, realizado pelo robô id_{Gbest}, é inicialmente verificado o número mínimo de bits β necessários para representar todos os identificadores t_i das tarefas definidas em $\mathbb{T} = \{t_1, t_2, \ldots, t_\tau\}$. Deste modo, β é definido pelo número de mínimo de bits da representação binária da tarefa de maior identificador, t_τ, como mostra a Tabela 1.

Tabela 1: Número mínimo de bits necessários para representar t_τ em $\mathbb{T}$

τ	$\mathbb{T}$	β
1	**0**	1
2	0,**1**	1
3	0,1,**2**	2
4	0,1,2,**3**	2
5	0,1,2,3,**4**	3
6	0,1,2,3,4,**5**	3
7	0,1,2,3,4,5,**6**	3
8	0,1,2,3,4,5,6,**7**	3
9	0,1,2,3,4,5,6,7,**8**	4
10	0,1,2,3,4,5,6,7,8,**9**	4
...	...	...
16	0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,**15**	4

É realizada a conversão do identificador de cada uma das tarefas da alocação $\mathbb{A}_{Gbest} = \{a_1, a_2, \ldots, a_\rho\}$ da representação decimal para a binária, sendo β o número de bits utilizado nesta conversão. Os valores convertidos são armazenados mantendo-se a ordem estabelecida pela alocação, para serem posteriormente divididos em 10 grupos de 8 bits cada. Cada grupo é convertido da representação binária para a decimal de forma a compor os 10 bytes da alocação $\mathbb{A}_{Gbest}$ codificada formando a carga útil para a mensagem à ser enviada pelo robô id_{Gbest}.

Quando a mensagem é recebida pelos demais robôs, a carga útil da mensagem é decodificada para compor a alocação $\mathbb{A}_{Gbest}$ informada pelo robô id_{Gbest}. Para realizar a decodificação, os 10 primeiros bytes da carga útil são convertidos da representação decimal para a binária, byte a byte, gerando 10 grupos com 8 bits cada. Os bits de cada grupo são armazenados, respeitando-se a ordem inicial dos bytes, para serem novamente separados em grupos binários de β bits. Cada grupo binário é convertido da representação binária

para a decimal compondo desta maneira as tarefas da alocação $\mathbb{A}_{Gbest}$.

5.4 Considerações Finais do Capítulo

Neste capítulo foram apresentados os aspectos da implementação dos algoritmos de alocação de tarefas ADTL e ADTG para robôs móveis do tipo Elisa III. O robô utilizado nos ensaios possui alguns dispositivos embarcados em uma estrutura simples de pequeno porte. Entre os dispositivos destaca-se o módulo de comunicação transmissor/receptor que possibilita a comunicação entre os robôs via rádio frequência, por intermédio de uma estação base ligada a um computador. Foram utilizados softwares na versão *open source* para a compilação e programação do robô. No capítulo 6 serão apresentados os resultados obtidos dos ensaios realizados com os robôs reais, sendo realizada uma análise dos mesmos.

Capítulo 6

ANÁLISE DOS RESULTADOS

NESTE capítulo são analisados os resultados obtidos pelas implementações dos algoritmos propostos no Capítulo 3 e Capítulo 4. Diferentes arranjos de enxame são utilizados nos ensaios com o objetivo de analisar a robustez dos algoritmos através dos resultados obtidos nestes ensaios. A análise dos resultados permite avaliar a eficácia dos algoritmos propostos, bem como a eficiência em atingir o objetivo principal de alocar as tarefas aos robôs do enxame, de modo que seja obtida uma nova alocação que respeite a proporção desejada. Após atingir a nova alocação adequada, o enxame deve manter-se nesta por tempo indeterminado até que uma nova proporção-objetivo seja informada.

A seção 6.1 apresenta a metodologia utilizada para realizar os ensaios. Na Seção 6.2 e Seção 6.3 são apresentados os resultados dos algoritmos ADTL e ADTG, respectivamente. Na Seção 6.4 é apresentada uma comparação entre os resultados obtidos. A Seção 6.5 apresenta as considerações finais para o capítulo.

6.1 Metodologia de Avaliação

A metodologia utilizada durante a avaliação dos resultados obtidos analisou diferentes arranjos de enxames, definindo o número de robôs ρ e tarefas τ. Os enxames foram inicializados com uma mesma alocação inicial $\mathbb{A}_0 = \{0, 0, \ldots, 0\}$, alocando a tarefa de identificador 0 a todos os robôs do enxame. Os enxames possuem o objetivo de alocar as tarefas entre os robôs de modo a atingir uma nova alocação $\mathbb{A}^*$ que respeite a proporção-objetivo $\mathbb{P}$.

Foram experimentados enxames de até 25 robôs com até 5 tarefas diferentes. Durante os ensaios, a escolha de ρ em cada enxame foi definida como sendo múltiplos de

quatro e cinco, compreendidos entre o intervalo de 4 a 25 robôs. A escolha de enxames com o número de robôs múltiplos de quatro, permitiu avaliar um possível impacto do protocolo de comunicação nos resultados, visto que o protocolo de comunicação otimiza a troca de mensagens de quatro robôs por vez, conforme mostrado no Capítulo 5. Nos ensaios, foram utilizados valores para τ compreendidos no intervalo de 2 a 5 tarefas, sendo estipulada uma proporção-objetiva para caso. Na Tabela 2 é apresentada a distribuição da proporção-objetivo entre as tarefas utilizadas para os ensaios realizados.

Tabela 2: Distribuição da proporção-objetivo entre as tarefas

# Tarefas	Proporção				
	Tarefa 0	Tarefa 1	Tarefa 2	Tarefa 3	Tarefa 4
2	60%	40%	–	–	–
3	20%	30%	50%	–	–
4	10%	15%	30%	45%	–
5	5%	10%	20%	30%	35%

Os ensaios realizados com enxames de diferentes arranjos de robôs e tarefas, fornecem casos de complexidades diferentes. Cada caso é definido através do arranjo do enxame, sendo caracterizados pelo par (ρ, τ). No Capítulo 1 foi analisada a complexidade do problema de alocação de tarefas para enxames homogêneos através do número de alocações factíveis $\#\mathbb{Q}$ geradas. Vale lembrar que, o valor de $\#\mathbb{Q}$ para enxames homogêneos é definido pelo número de tarefas τ elevado ao número de robôs ρ presentes no enxame.

A Tabela 3 apresenta o número de alocações factíveis $\#\mathbb{Q}$ para cada ensaio realizado. Nota-se que a complexidade aumenta com o aumento de ρ e τ.

Tabela 3: Número de alocações factíveis $\#\mathbb{Q}$ para os ensaios realizados

$\#\mathbb{Q}$	$\tau = 2$	$\tau = 3$	$\tau = 4$	$\tau = 5$
$\rho = 4$	16	81	256	625
$\rho = 5$	32	243	1024	3125
$\rho = 8$	256	6561	65536	390625
$\rho = 10$	1024	59049	1048576	9765625
$\rho = 12$	4096	531441	16777216	244140625
$\rho = 15$	32768	14348907	1073741824	30517578125
$\rho = 16$	65536	43046721	4294967296	$1,52588 \times 10^{11}$
$\rho = 20$	1048576	3486784401	$1,09951 \times 10^{12}$	$9,53674 \times 10^{13}$
$\rho = 24$	16777216	$2,8243 \times 10^{11}$	$2,81475 \times 10^{14}$	$5,96046 \times 10^{16}$
$\rho = 25$	33554432	$8,47289 \times 10^{11}$	$1,1259 \times 10^{15}$	$2,98023 \times 10^{17}$

A avaliação de cada um dos 40 diferentes pares (ρ, τ) formados, conforme mostrado na Tabela 3, foi executada por 10 vezes, totalizando 400 diferentes ensaios para cada um dos algoritmos propostos. A partir dos resultados obtidos, foram avaliados os aspectos relacionados à convergência e comunicação do enxame durante a alocação de tarefas entre os robôs até que o enxame alcance uma alocação $\mathbb{A}^*$ que respeite a proporção-objetivo $\mathbb{P}$ prescrita.

Como aspectos de convergência, foram analisados se os algoritmos alcançaram o objetivo principal, i.e. encontrar $\mathbb{A}^*$, e em caso positivo, o tempo em milissegundos (ms) decorrido entre o inicio e término do processo de alocação. Considerou-se que o algoritmo alcançou o seu objetivo principal ao alocar o enxame para uma nova alocação $\mathbb{A}^*$, de forma que a alocação se mantivesse inalterada por um longo período de tempo. Exclusivamente para o algoritmo ADTG, que utiliza o PSO, foi analisado também o número de iterações realizadas até o enxame alcançar o seu objetivo principal.

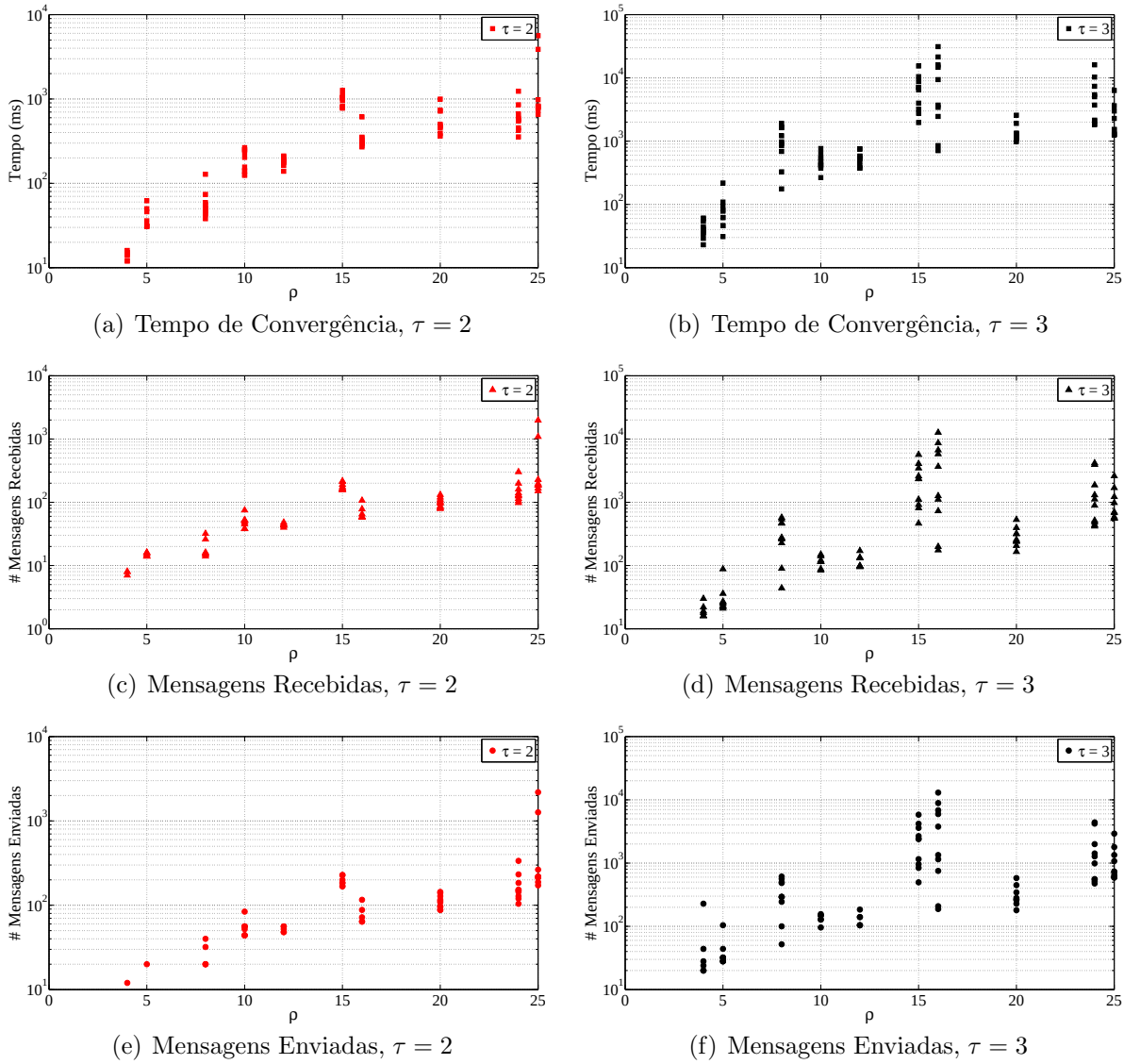
Para a análise dos aspectos de comunicação, foram observados os números de mensagens trocadas entre os robôs durante os processos de comunicação. Como forma de referenciar a troca de mensagens, foi estabelecido que as mensagens provenientes da estação-base com destino aos robôs são definidas como *Mensagens Enviadas* e as mensagens destinadas à estação-base e geradas pelos robôs são definidas como *Mensagens Recebidas*.

6.2 Resultados obtidos pelo ADTL

Com o objetivo de avaliar o desempenho do algoritmo ADTL, apresentado no Capítulo 3, são realizados ensaios variando-se o número de tarefas e robôs do enxame. Para cada caso, foram realizados 10 diferentes ensaios, sendo os resultados obtidos apresentados na Figura 21, para os ensaios dos casos com $\tau = 2$ e $\tau = 3$, e Figura 22, para os ensaios dos casos com $\tau = 4$ e $\tau = 5$.

Os gráficos apresentados na Figura 21 e Figura 22 representam os resultados para os 10 ensaios de cada caso analisado, variando-se o número de robôs ρ do enxame de 4 a 25. Os gráficos permitem avaliar, localmente, a dispersão dos resultados obtidos dentro de cada caso analisado e, globalmente, a tendência dos resultados entre os casos.

A dispersão serve para avaliar o quanto os dados são semelhantes entre si, des-



(a) Tempo de Convergência, $\tau = 2$

(b) Tempo de Convergência, $\tau = 3$

(c) Mensagens Recebidas, $\tau = 2$

(d) Mensagens Recebidas, $\tau = 3$

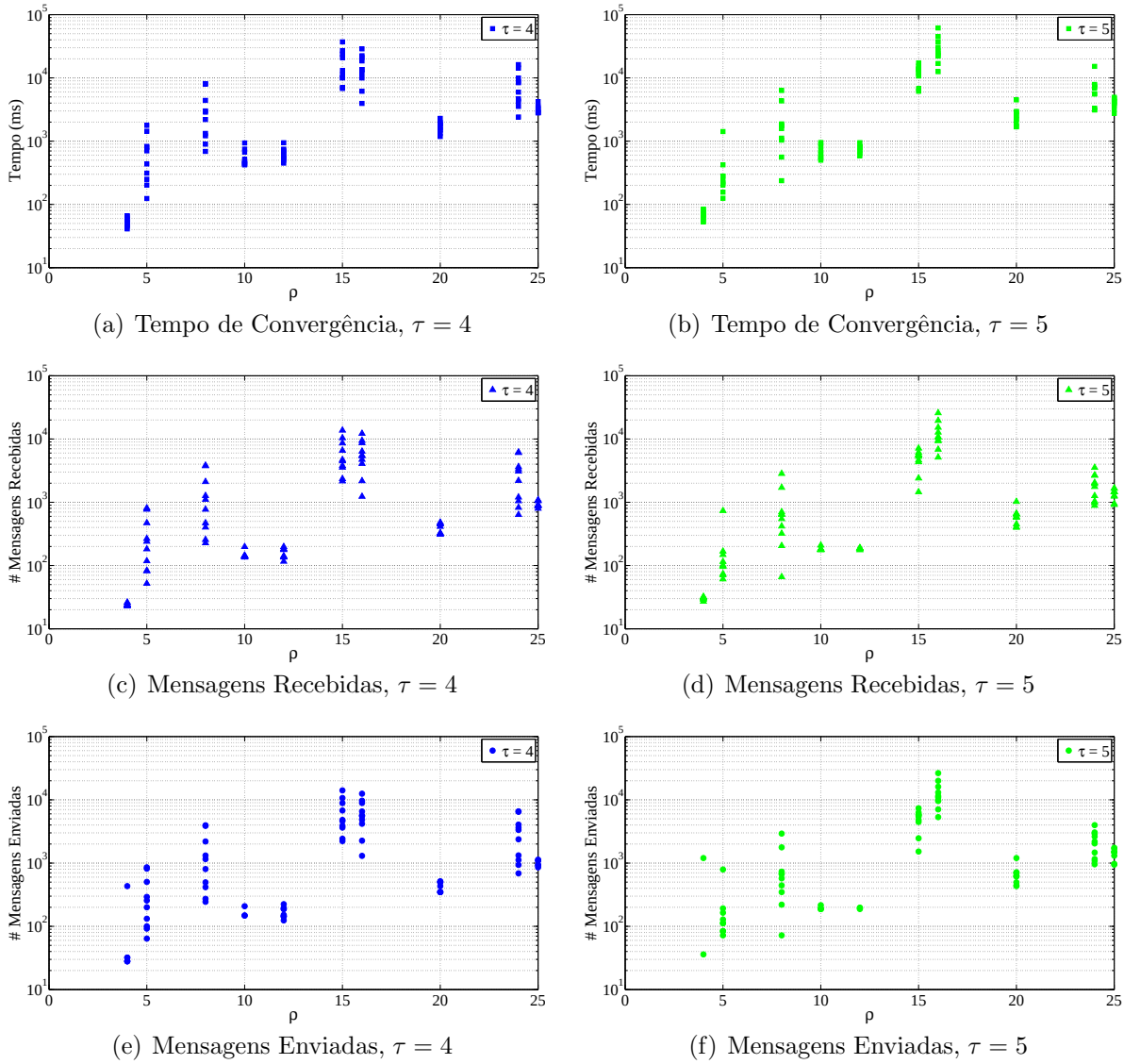
(e) Mensagens Enviadas, $\tau = 2$

(f) Mensagens Enviadas, $\tau = 3$

Figura 21: Resultado dos ensaios realizados para $\tau = 2$ e $\tau = 3$ em ADTL

crevendo assim o quanto os dados se distanciam do valor central. Nesta avaliação, o valor central é representado pela média dos resultados obtidos. Na avaliação local, como exemplo ilustrativo são comparados os casos (4,4) e (5,4) do gráfico que avalia o tempo de convergência na Figura 22(a). No caso (4,4), os resultados obtidos possuem valores com menor dispersão da média dos resultados enquanto que no caso (5,4), os resultados obtidos possuem valores com maior dispersão da média dos resultados.

Para os casos onde os ensaios apresentaram resultados com maior dispersão, foram realizados novos ensaios com o objetivo de comparar os resultados destes novos ensaios com os anteriores. Esta comparação buscou verificar se a dispersão apresentada pelos resultados iniciais foi causada por uma influência externa, tal como uma elevada perda de



(a) Tempo de Convergência, $\tau = 4$

(b) Tempo de Convergência, $\tau = 5$

(c) Mensagens Recebidas, $\tau = 4$

(d) Mensagens Recebidas, $\tau = 5$

(e) Mensagens Enviadas, $\tau = 4$

(f) Mensagens Enviadas, $\tau = 5$

Figura 22: Resultado dos ensaios realizados para $\tau = 4$ e $\tau = 5$ em ADTL

mensagens ocorrida durante o processo de comunicação. Desta forma, foram realizados 5 novos ensaios para estes casos de interesse para que os novos resultados obtidos fossem comparados aos resultados iniciais, quanto à dispersão. Para as situações onde os resultados iniciais possuem uma dispersão menor que os novos resultados, os resultados das simulações iniciais são mantidos. Caso contrário, os resultados inicias mais dispersos da média são substituídos pelos novos resultados menos dispersos.

Na avaliação global, os gráficos mostrados na Figura 21 e na Figura 22, de forma geral, apresentam uma tendência de crescimento em decorrência do aumento do número de robôs no enxame. Esta tendência é melhor observada nos gráficos dos casos realizados para duas tarefas ($\tau = 2$), mostrados na Figura 21(a), Figura 21(c) e Figura 21(e).

Para avaliar a comunicação, os resultados apresentaram, de modo geral, uma distribuição parecida ao comparar os gráficos de números de mensagens recebidas e enviadas para cada grupo de ensaios com o mesmo número de tarefas. Para os gráficos dos ensaios realizados com $\tau = 2$, exibidos na Figura 21(c) e Figura 21(e) é possível observar que os resultados obtidos nos diferentes enxames apresentam uma certa similaridade. Esta observação é confirmada pela estrutura de troca de mensagens implementada, conforme descrito no Capítulo 5.

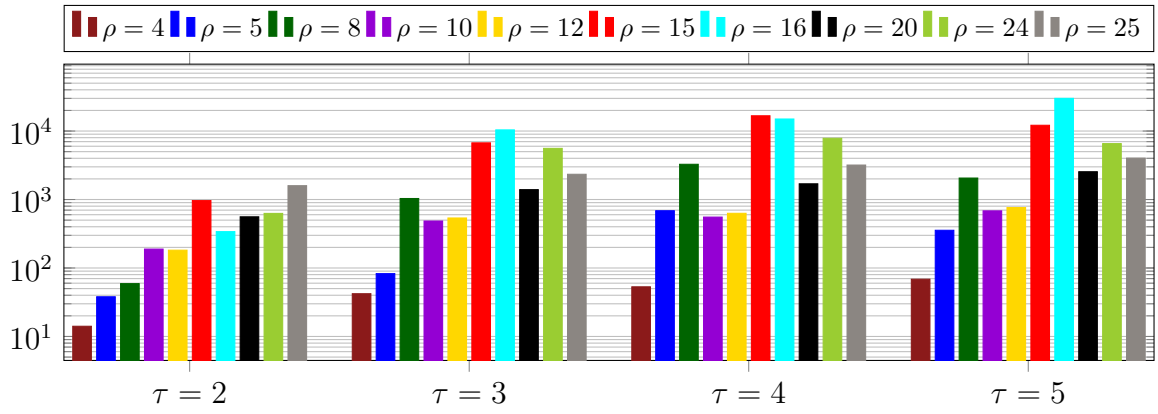


Figura 23: Média dos resultados de tempo de convergência (ms) para o algoritmo ADTL

Na Figura 23 são apresentadas as médias dos resultados obtidos para o tempo de convergência, em milissegundos, para o algoritmo ADTL em cada caso, mostrados anteriormente nos gráficos da Figura 21(a) para $\tau = 2$, Figura 21(b) para $\tau = 3$, Figura 22(a) para $\tau = 4$ e Figura 22(b) para $\tau = 5$.

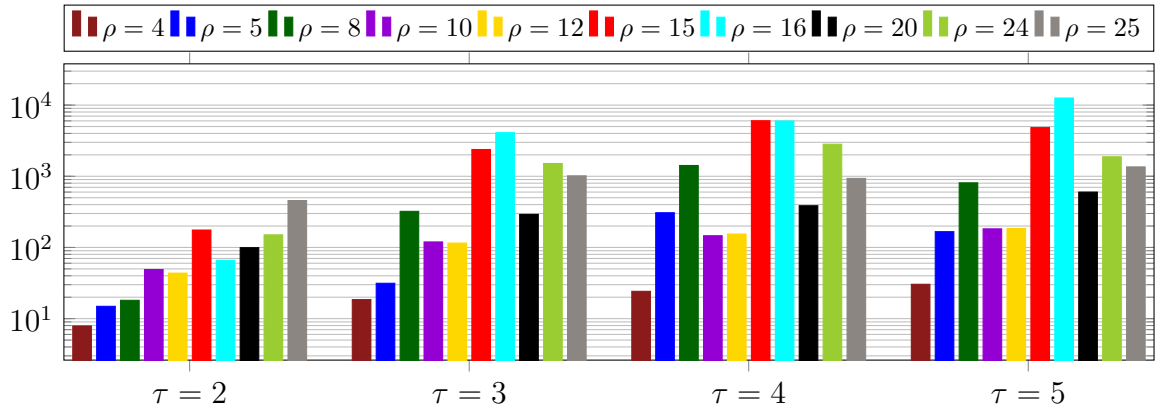


Figura 24: Média dos Resultados de número de Mensagens recebidas para o algoritmo ADTL

Na Figura 24 são apresentadas as médias dos resultados obtidos para o número de mensagens recebidas, para o algoritmo ADTL em cada caso, mostrados nos gráficos da Figura 21(c) para $\tau = 2$, Figura 21(d) para $\tau = 3$, Figura 22(c) para $\tau = 4$ e Figura 22(d) para $\tau = 5$.

Na Figura 25 são apresentadas as médias dos resultados obtidos para o número de mensagens enviadas, para o algoritmo ADTL em cada caso, mostrados nos gráficos da Figura 21(e) para $\tau = 2$, Figura 21(f) para $\tau = 3$, Figura 22(e) para $\tau = 4$ e Figura 22(f) para $\tau = 5$.

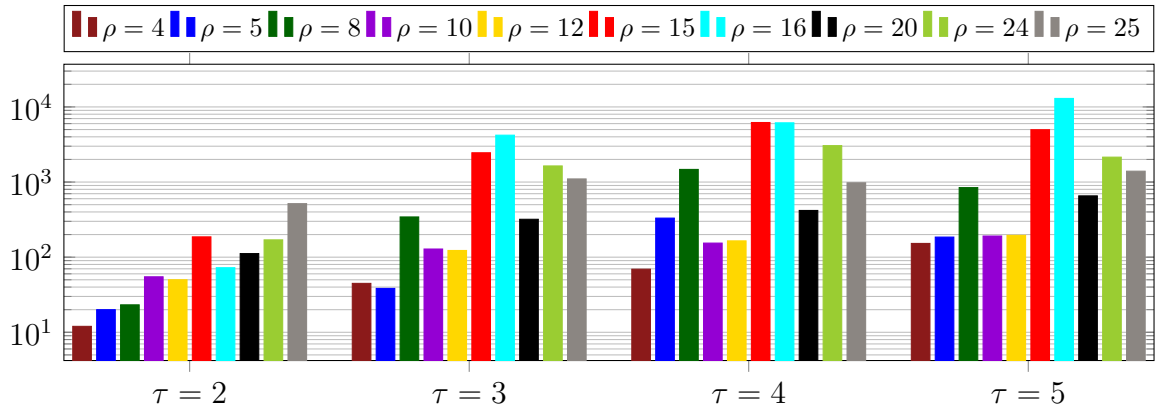
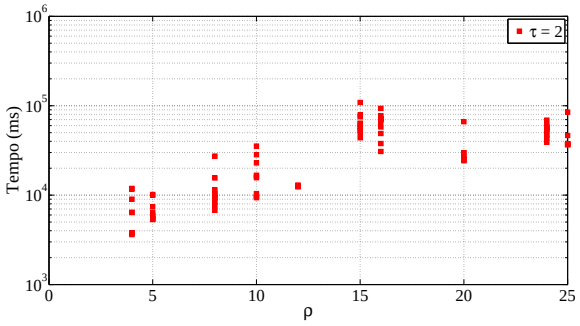


Figura 25: Média dos Resultados de número de mensagens enviadas para o algoritmo ADTL

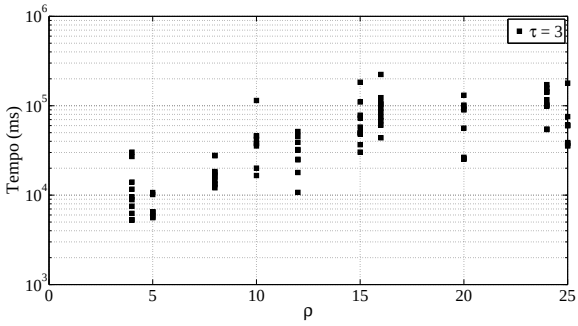
6.3 Resultados obtidos pelo ADTG

Com o objetivo de avaliar o desempenho do algoritmo ADTG, apresentado no Capítulo 4, são realizados ensaios variando-se o número de tarefas e robôs do enxame, da mesma forma que explicado na seção anterior para o algoritmo ADTL. Para cada caso, foram realizados 10 diferentes ensaios, sendo os resultados obtidos apresentados na Figura 26, para os ensaios dos casos com $\tau = 2$ e $\tau = 3$, e Figura 27, para os ensaios dos casos com $\tau = 4$ e $\tau = 5$.

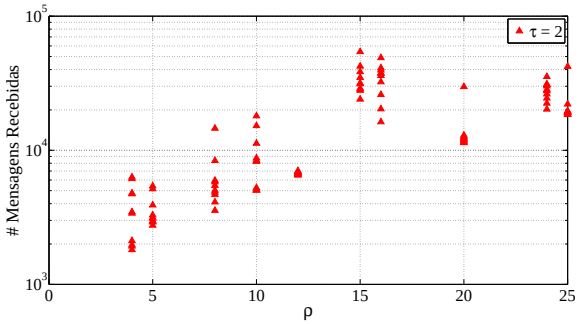
Os gráficos apresentados na Figura 26 e Figura 27 representam os resultados para os 10 ensaios realizados em cada caso analisado variando o número de robôs ρ do enxame de 4 a 25. Os gráficos permitem avaliar os aspectos locais de dispersão dos resultados obtidos em cada caso e os aspectos globais de tendência dos resultados entre os casos estudados.



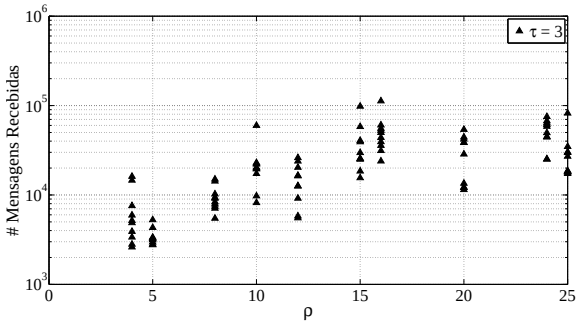
(a) Tempo de Convergência, $\tau = 2$



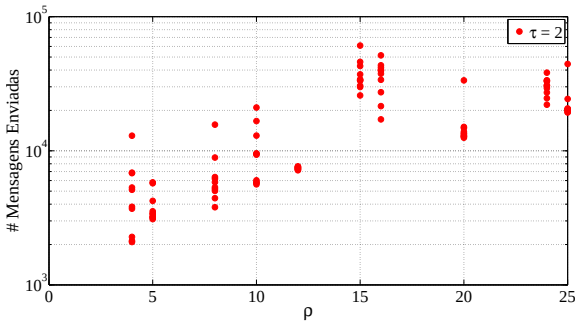
(b) Tempo de Convergência, $\tau = 3$



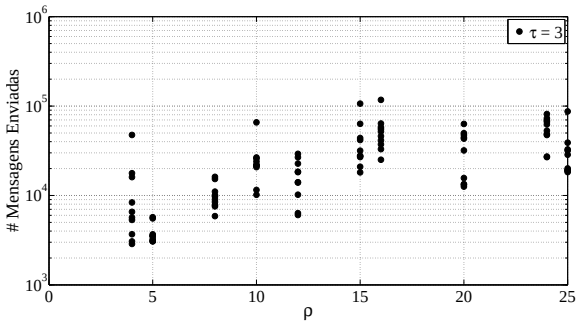
(c) Mensagens Recebidas, $\tau = 2$



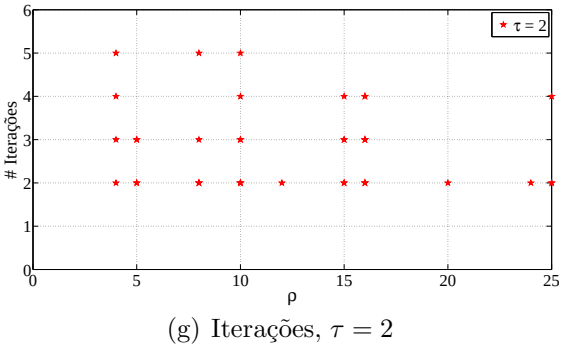
(d) Mensagens Recebidas, $\tau = 3$



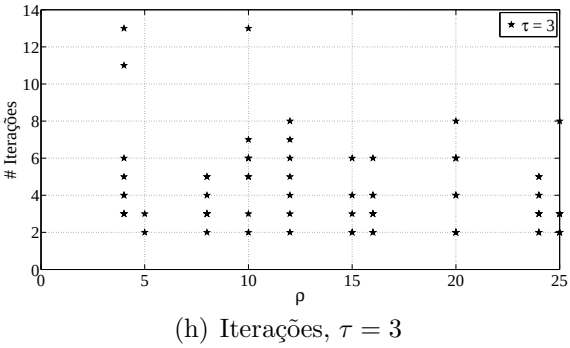
(e) Mensagens Enviadas, $\tau = 2$



(f) Mensagens Enviadas, $\tau = 3$



(g) Iterações, $\tau = 2$



(h) Iterações, $\tau = 3$

Figura 26: Resultado dos ensaios realizados para $\tau = 2$ e $\tau = 3$ em ADTG

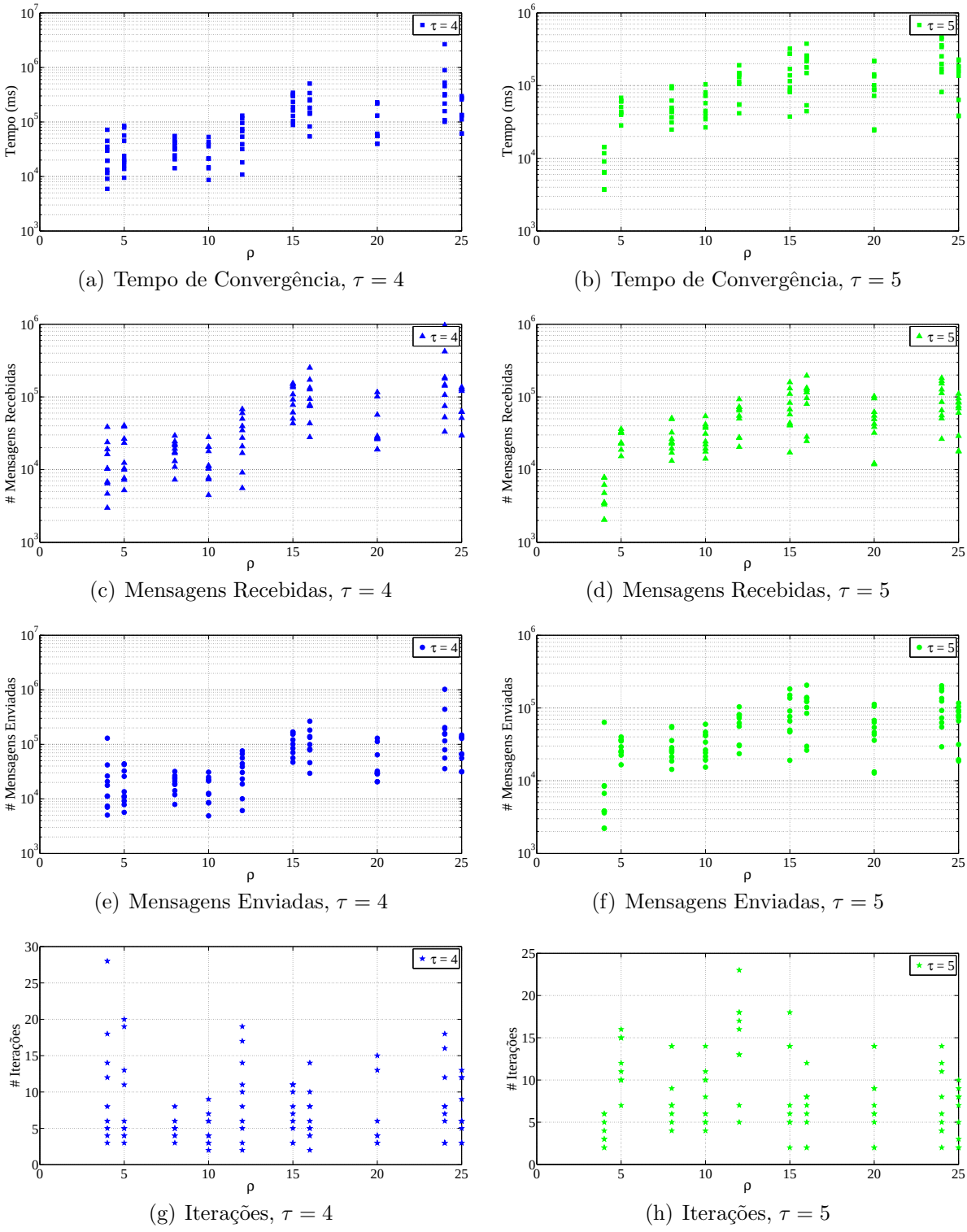
De maneira análoga ao realizado durante a avaliação dos aspectos locais de ADTL, foi analisada a dispersão dos resultados obtidos em relação à média. Para os casos de interesse, onde os ensaios apresentaram resultados com maior dispersão, foram realizados novos ensaios com o objetivo de comparar os resultados destes novos ensaios com os anteriores. Para cada caso de interesse foram realizados 5 novos ensaios, permitindo que os novos resultados gerados fossem comparados com os resultados inicialmente obtidos. Nas situações em que os resultados iniciais apresentaram uma dispersão em relação à média com valor menor que a dispersão apresentada pelos novos resultados, os resultados iniciais são mantidos. Contudo, nas situações em que o valor da dispersão dos resultados iniciais é superior ao valor de dispersão dos novos resultados, os resultados obtidos nos ensaios iniciais são substituídos pelos resultados dos novos ensaios.

Para a avaliação do aspecto global, os gráficos mostrados na Figura 26 e na Figura 27 apresentam uma tendência de aumento ao longo do aumento do número de robôs no enxame. Esta tendência é melhor observada nos gráficos dos casos realizados para quatro tarefas ($\tau = 4$), mostrados na Figura 27(a), Figura 27(c) e Figura 27(e). Assim, é possível notar que mesmo com o fato dos gráficos mencionados apresentarem uma elevada dispersão em seus resultados, a partir do aspecto global é possível notar claramente a tendência de crescimento em consequência ao aumento no número de robôs no enxame.

Quanto aos aspectos de comunicação, os resultados apresentaram uma distribuição similar ao comparar os gráficos dos números de mensagens recebidas e enviadas de cada grupo dos ensaios. Para os gráficos dos ensaios realizados com $\tau = 2$, exibidos na Figura 26(c) e Figura 26(e) é possível observar que os resultados obtidos nos diferentes enxames apresentam uma certa similaridade entre eles.

Os resultados referentes ao número de iterações realizadas até o enxame convergir para a alocação $\mathbb{A}^*$ são mostrados nos gráficos apresentados na Figura 26(g) e Figura 26(h), para os ensaios dos casos com $\tau = 2$ e $\tau = 3$, e Figura 27(g) e Figura 27(h), para os ensaios dos casos com $\tau = 4$ e $\tau = 5$.

A Figura 28 apresenta as médias dos resultados obtidos para o tempo de convergência, em milissegundos, para o algoritmo ADTG em cada caso analisado, mostrados anteriormente nos gráficos da Figura 26(a) para $\tau = 2$, Figura 26(b) para $\tau = 3$, Figura 27(a) para $\tau = 4$ e Figura 27(b) para $\tau = 5$.



(a) Tempo de Convergência, $\tau = 4$

(b) Tempo de Convergência, $\tau = 5$

(c) Mensagens Recebidas, $\tau = 4$

(d) Mensagens Recebidas, $\tau = 5$

(e) Mensagens Enviadas, $\tau = 4$

(f) Mensagens Enviadas, $\tau = 5$

(g) Iterações, $\tau = 4$

(h) Iterações, $\tau = 5$

Figura 27: Resultado dos ensaios realizados para $\tau = 4$ e $\tau = 5$ em ADTG

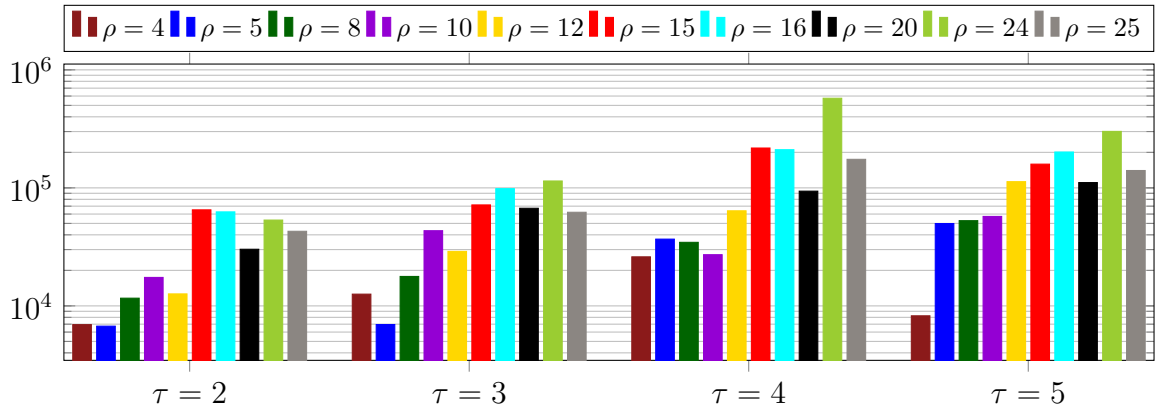


Figura 28: Média dos resultados de tempo de convergência (ms) para o algoritmo ADTG

A Figura 29 apresenta as médias dos resultados obtidos para o número de mensagens recebidas, para o algoritmo ADTG em cada caso analisado, mostrados anteriormente nos gráficos da Figura 26(c) para $\tau = 2$, Figura 26(d) para $\tau = 3$, Figura 27(c) para $\tau = 4$ e Figura 27(d) para $\tau = 5$.

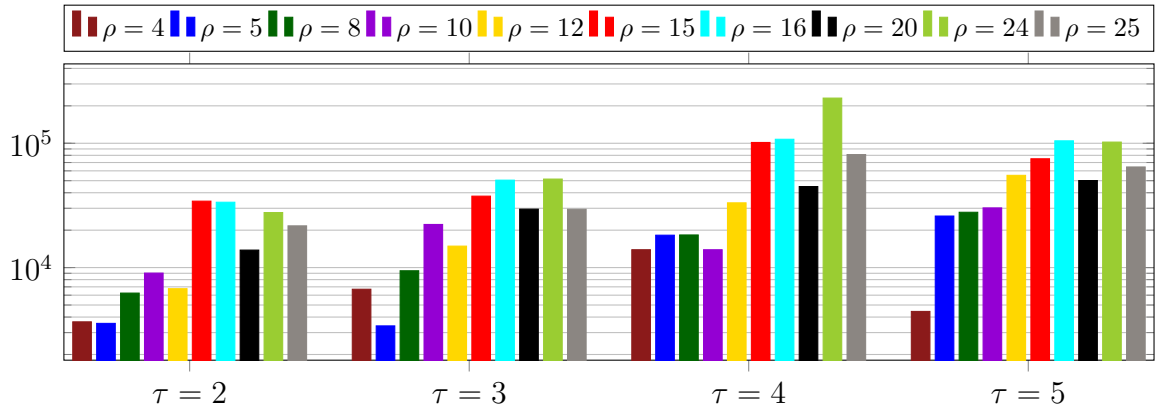


Figura 29: Média dos Resultados de número de mensagens recebidas para o algoritmo ADTG

A Figura 30 apresenta as médias dos resultados obtidos para o número de mensagens enviadas, para o algoritmo ADTG em cada caso analisado, mostrados nos gráficos da Figura 26(e) para $\tau = 2$, Figura 26(f) para $\tau = 3$, Figura 27(e) para $\tau = 4$ e Figura 27(f) para $\tau = 5$.

Na Figura 31 é apresentado o diagrama com a média do número de iterações utilizadas para o algoritmo ADTG em cada caso analisado, mostrados nos gráficos da Figura 26(g) para $\tau = 2$, Figura 26(h) para $\tau = 3$, Figura 27(g) para $\tau = 4$ e Figura 27(h)

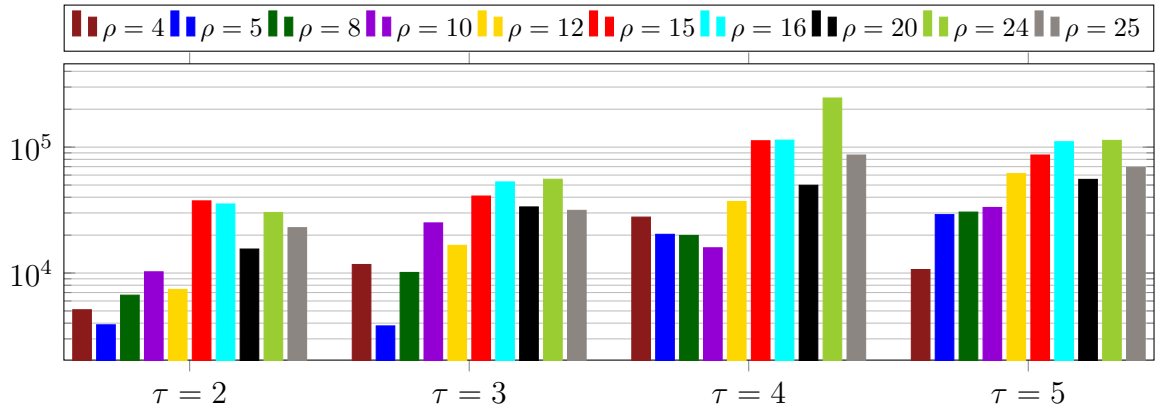


Figura 30: Média dos Resultados de número de mensagens enviadas para o algoritmo ADTG

para $\tau = 5$, até o mesmo convergir para a proporção-objetivo. Verifica-se a partir da Figura que o número de iterações aumenta de acordo com o número de tarefas, de modo que para os casos com um maior número de tarefas é necessário um maior número de iterações.

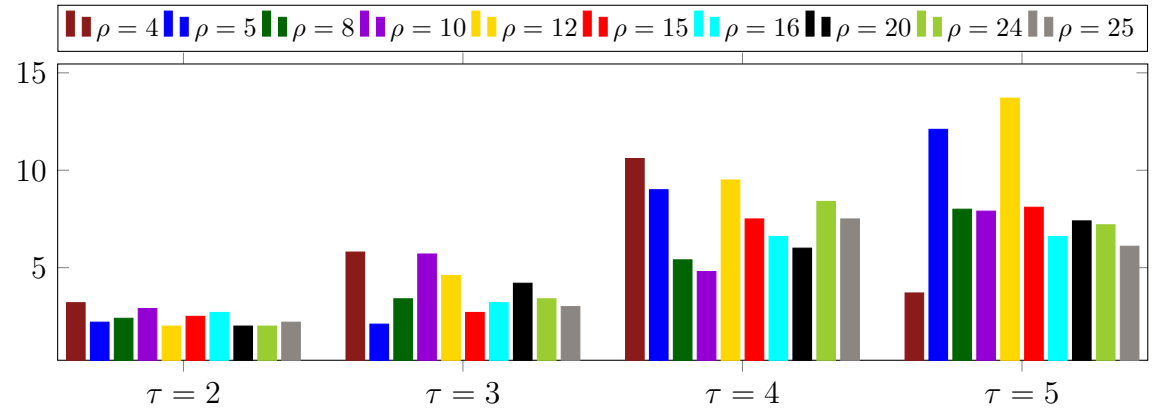


Figura 31: Média dos resultados de número de iterações para o algoritmo ADTG

6.4 Comparação entre os algoritmos propostos

Nesta Seção são apresentadas as comparações entre os resultados obtidos durante os ensaios realizados pelos algoritmos ADTL e ADTG, mostrados na Seção 6.2 e Seção 6.3, respectivamente.

A partir dos diagramas das médias de tempo de convergência, apresentados na Figura 23 para ADTL e na Figura 28 para ADTG, é apresentada a Figura 32 com os re-

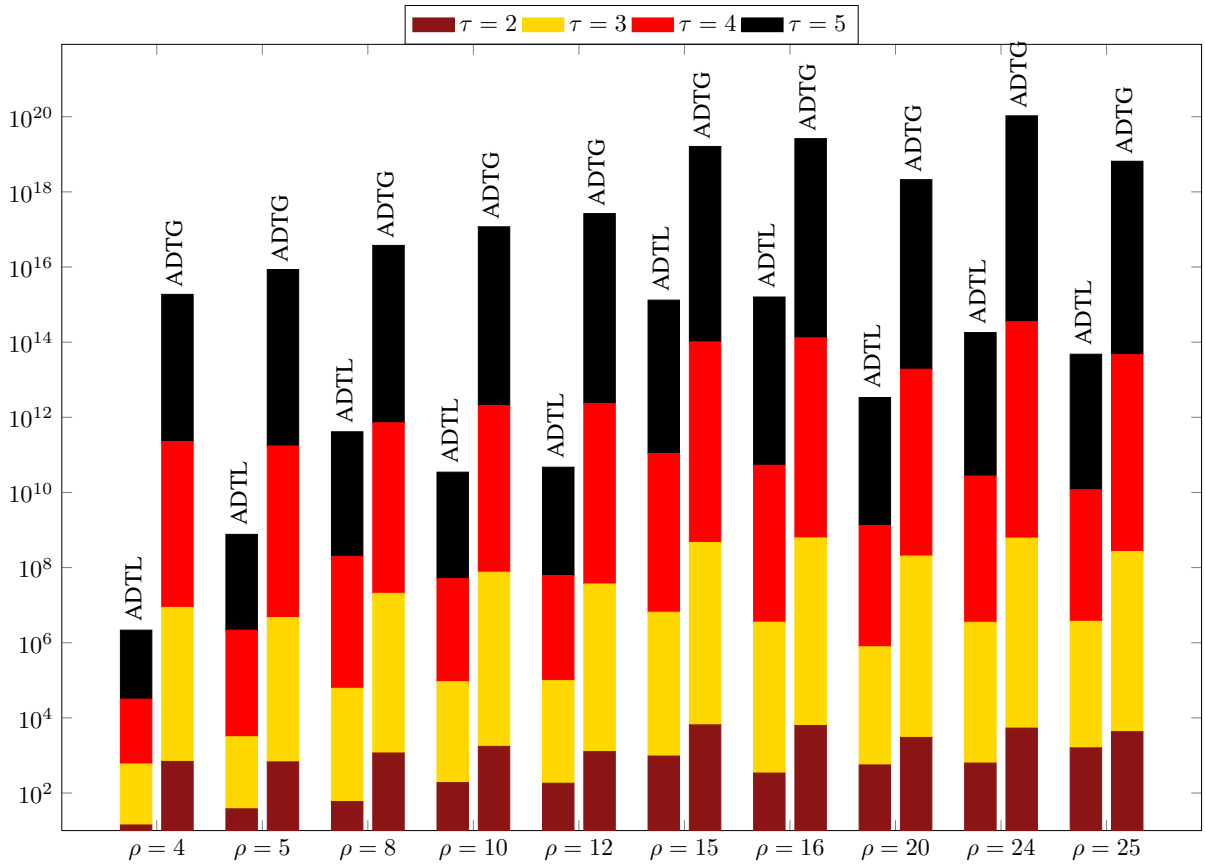


Figura 32: Resultados de tempo de convergência (ms) para ADTL e ADTG

sultados de média para os algoritmos. Pode-se constatar que em todos os casos analisados o tempo de convergência para a proporção-objetivo apresentou uma tendência de aumento conforme o aumento no número de robôs e tarefas. Pode ser observado ainda, que em todos os casos os resultados de tempo de convergência para o algoritmo ADTL foram inferiores aos resultados obtidos para o algoritmo ADTG. Dessa forma, o algoritmo ADTL converge mais rapidamente para a proporção objetivo quando comparado ao algoritmo ADTG.

Comparando-se as duas implementações, constata-se que no caso (4,2) o algoritmo ADTL apresentou a maior diferença no tempo de convergência em relação ao algoritmo ADTG, chegando a ser 492,13 vezes mais rápido, enquanto que no caso (16,5) foi apresentada a menor diferença, sendo ADTL 6,7 vezes mais rápido que ADTG. Observa-se ainda, que nos casos para duas tarefas ($\tau = 2$) a diferença do tempo de convergência entre os algoritmos tende a ser maior que nos demais casos com um maior número de tarefas, ou seja, para um enxame com um mesmo número de robôs um aumento no número de

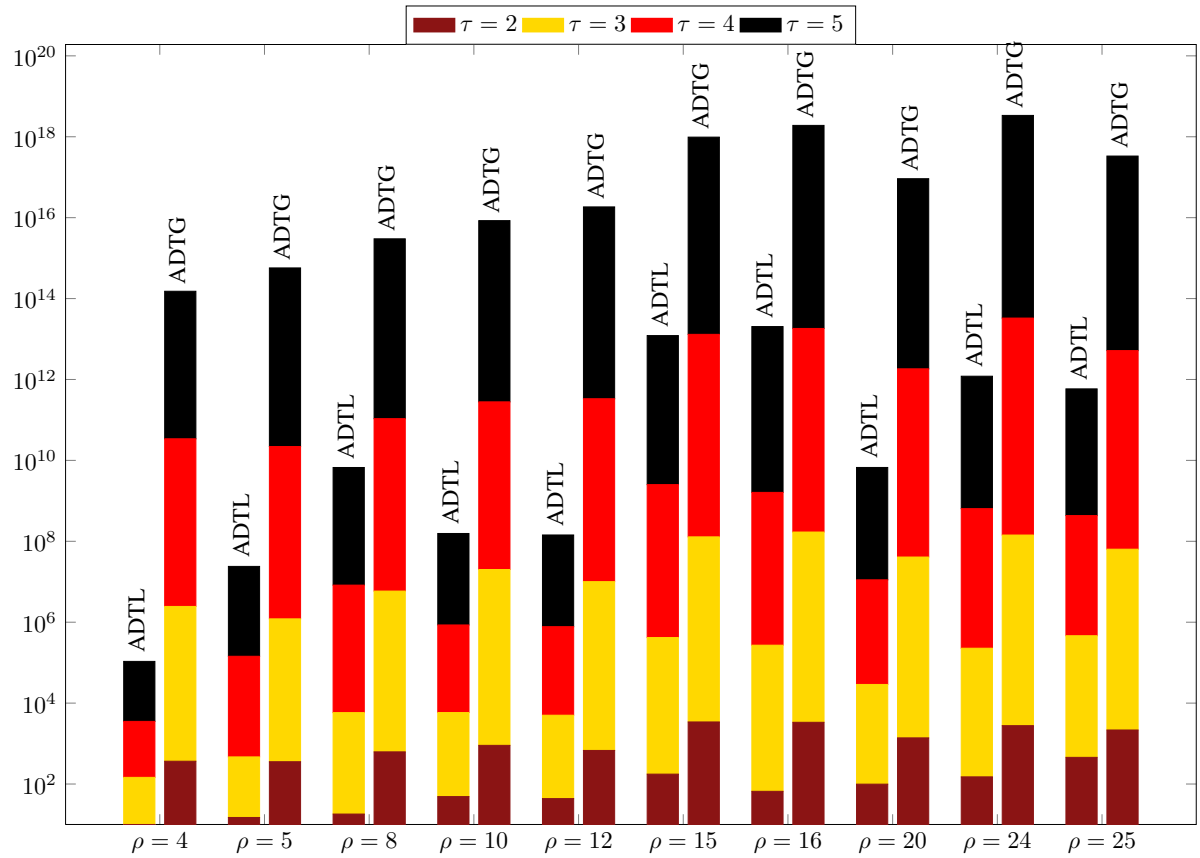


Figura 33: Resultados de número de mensagens recebidas para ADTL e ADTG

tarefas indica que a rapidez de ADTL em relação a ADTG tende a diminuir. Apenas o caso (12,2), sendo 69,4 vezes mais rápido, e o caso (25,2), sendo 26,93 vezes mais rápido, mostraram ser exceções desta regra, de modo que os casos (12,5) e (25,4), respectivamente, mostraram ser mais rápidos para um mesmo número de robôs.

A partir dos diagramas referentes à média do número de mensagens recebidas, apresentados na Figura 24 para ADTL e na Figura 29 para ADTG, é apresentada a Figura 33 com os resultados de média para os algoritmos. Podemos constatar que o numero de mensagens recebidas durante o processo de comunicação para o algoritmo ADTL é menor quando comparado ao algoritmo ADTG em todos os casos. Os casos extremos desta observação, são o caso (16,2) que apresentou a maior diferença possuindo 508,41 vezes menos mensagens recebidas que ADTG e o caso (16,5), que apresentou a menor diferença possuindo apenas 8,32 vezes menos mensagens recebidas que ADTG.

A partir dos diagramas referentes a média do número de mensagens enviadas, apresentados na Figura 25 para ADTL e na Figura 30 para ADTG, é apresentada a

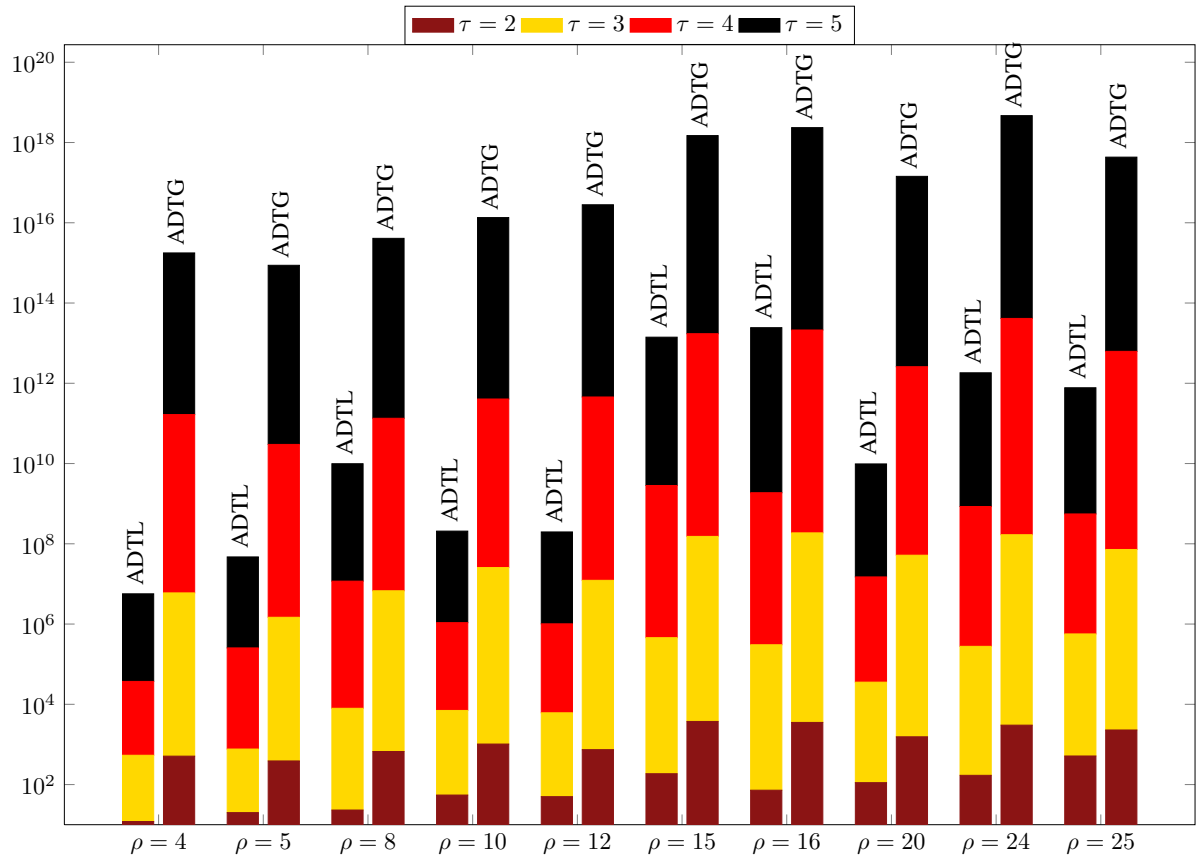


Figura 34: Resultados de número de mensagens enviadas para ADTL e ADTG

Figura 34 com os resultados de média para os algoritmos. Podemos constatar que o numero de mensagens enviadas durante o processo de comunicação para o algoritmo ADTL é menor quando comparado ao algoritmo ADTG em todos os casos. São relatados o caso (16,2), que apresentou a maior diferença possuindo 488,49 vezes menos mensagens enviadas que ADTG, e o caso (16,5), que apresentou a menor diferença possuindo apenas 8,32 vezes menos mensagens enviadas que ADTG.

Dessa forma, ao analisarmos a Figura 24 e Figura 25, referentes ao número de mensagens trocadas no processo de comunicação para ADTL, e a Figura 29 e Figura 30, referentes ao número de mensagens trocadas no processo de comunicação para ADTG, podemos constatar que o número de mensagens trocadas entre os robôs em ADTL é menor que em ADTG. Esta constatação é decorrente da implementação dos algoritmos propostos, de modo que o algoritmo ADTL necessita de um número menor de mensagens para convergir para a proporção objetivo quando comparado ao algoritmo ADTG. Por consequência, o tempo de convergência é superior no algoritmo ADTG em relação ao

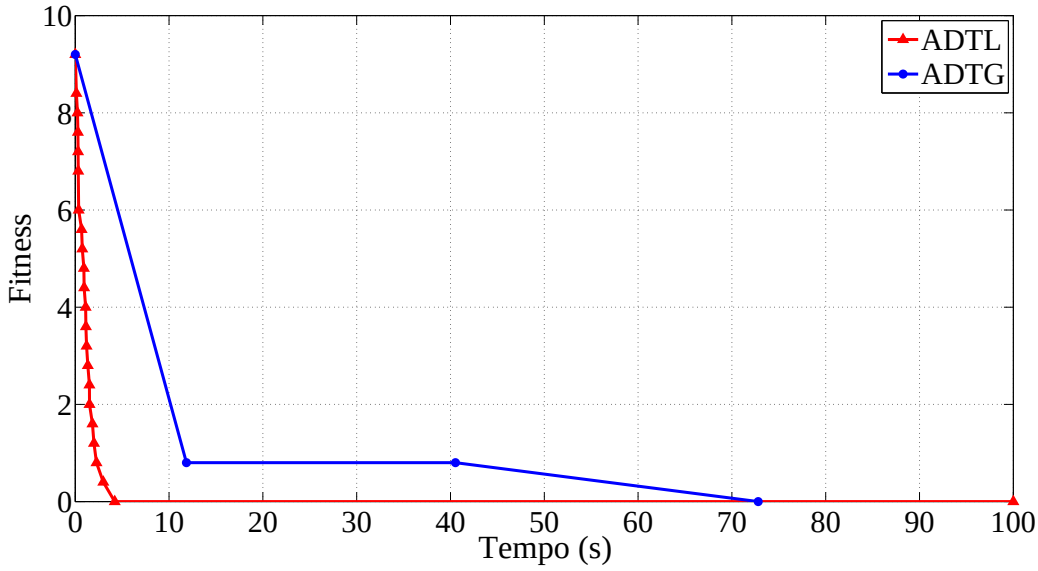


Figura 35: Fitness ao longo do tempo (segundos) para um enxame de 25 robôs com 5 tarefas à serem alocadas utilizando os algoritmos ADTL e ADTG

algoritmo ADTL.

Nota-se no processo de comunicação que a relação entre o número de mensagens enviadas e recebidas de um mesmo caso é muito próxima, sendo o número de mensagem enviadas um pouco superior ao de recebidas, em torno de uma a duas vezes maior na maior parte dos casos. Esta relação é esperada uma vez que a implementação da comunicação utilizada pelos algoritmos busca realizar a troca de mensagens através de envio e confirmação.

Nos casos com 4 robôs, para ambos os algoritmos, podemos perceber que a relação tornou-se um pouco maior que o esperado. Seja o caso (4,5) para ADTL, que possui relação de 5 vezes, e o caso (4,5) para ADTG, que possui relação de 2,4 vezes, os casos relatados com o maior número de mensagens enviadas em relação ao de recebidas. Esta situação é, possivelmente, decorrente de uma perda excessiva de mensagens durante o processo de comunicação.

Na Figura 35 é apresentada uma comparação entre os algoritmos propostos através de dois ensaios realizados para um enxame de 25 robôs e 5 tarefas. O gráfico mostra a relação do fitness $f(\mathbb{A})$ obtido ao longo do tempo, em segundos, para cada um dos algoritmos.

Observa-se que o ensaio realizado com o algoritmo ADTL alterou a alocação dos

robôs ao longo do ensaio até atingir a proporção objetivo em $f(\mathbb{A}) = 0$ mais rapidamente que o algoritmo ADTG. Dessa forma, podemos constatar que os algoritmos possuem a característica de alterar a alocação do enxame no sentido de convergir gradativamente para uma alocação que atenda a proporção objetivo desejada.

A Figura 36 apresenta os resultados obtidos do fitness $f(\mathbb{A})$ ao longo do tempo, em segundos, para os dois ensaios ($ADTL_1$ e $ADTG$) apresentados na Figura 35 e para um terceiro ensaio ($ADTL_2$) utilizando o algoritmo ADTL. Os três ensaios são realizados para um enxame de 25 robôs e 5 tarefas iniciado em uma alocação de tarefas $\mathbb{A}_0$ e que busca atingir uma nova alocação $\mathbb{A}$ de forma que $f(\mathbb{A}) = 0$.

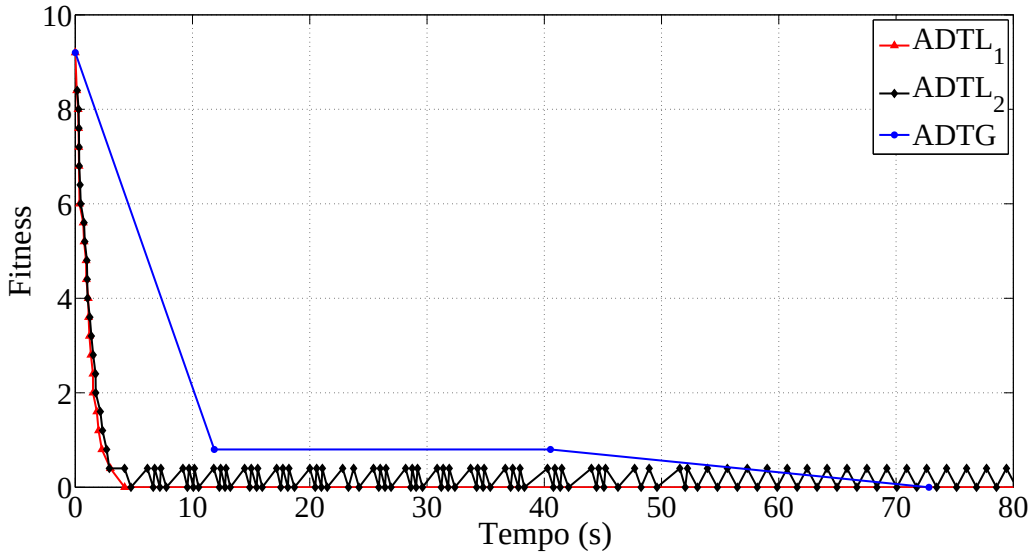


Figura 36: Fitness ao longo do tempo (segundos) para um enxame de 25 robôs com 5 tarefas à serem alocadas utilizando o algoritmo ADTL, em dois ensaios, e o algoritmo ADTG

O gráfico mostra que o ensaio para o enxame $ADTL_2$ alcança pela primeira vez uma alocação $\mathbb{A}^*$ após 5 segundos do início do ensaio. Entretanto, a alocação do enxame não é mantida fazendo com que o algoritmo seja obrigado a intervir novamente para altera-la. Dessa forma, $ADTL_2$ não consegue realizar a alteração da alocação do enxame de modo que a mesma se mantenha estável ao longo do tempo observado. Esta constatação é relatada em alguns ensaios realizados para ADTL verificando-se uma instabilidade na ação de manter a alocação de tarefas após o enxame atingir a convergência.

6.5 Considerações Finais do Capítulo

Neste capítulo foram apresentados os resultados para os ensaios realizados com os algoritmos ADTL e ADTG para diferentes arranjos de enxame. Nos resultados obtidos, verificou-se a superioridade do algoritmo ADTL em relação à ADTG na questão do tempo de convergência.

Nos aspectos de comunicação, o algoritmo ADTL mostrou necessitar de um número menor de mensagens trocadas para que o enxame atingisse a proporção objetivo, quando comparado à ADTG. Contudo, observou-se em alguns ensaios realizados no algoritmo ADTL a ocorrência de instabilidade em manter a alocação do enxame após o mesmo atingir a convergência.

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 7

CONCLUSÕES E TRABALHOS FUTUROS

ALOCAÇÃO dinâmica de tarefas é um processo necessário para o bom gerenciamento do enxame. Os algoritmos de alocação de tarefas possuem o objetivo de executar uma tarefa complexa decompondo-a em tarefas simples que são facilmente executadas pelos indivíduos do enxame. A realização coordenada destas simples tarefas entre os indivíduos do enxame, de modo que seja respeitada uma proporção pré-definida de execução, permite a realização da tarefa complexa.

7.1 Conclusões

Esta dissertação abordou o problema de alocação dinâmica de tarefas para um enxame de robôs. A escolha deste tema foi motivado pelo crescente interesse em sua utilização na solução de problemas de elevada complexidade, sendo assim um campo de pesquisa amplamente em expansão. A aplicação de algoritmos de ADT à robótica móvel permitiu realização de atividades antes incompatíveis ao trabalho humano. Diversas aplicações usando a robótica de enxame, requerem uma alocação dinâmica de tarefas. Por exemplo, em situações que representem um risco ou inviabilizem a presença humana, o enxame de robôs seria capaz de se auto-organizar em grupos, sendo cada grupo designado a realizar uma tarefa específica. A realização das tarefas pelos robôs de modo coordenado permite que o objetivo seja alcançado e o enxame realize uma ação significativa e complexa.

Foi realizada uma análise quanto a complexidade do processo de alocação de tarefas no Capítulo 1. Demonstrou-se que o número de alocações factíveis $\#\mathbb{Q}$ é maior

para os enxames homogêneos do que para enxames heterogêneos. Foi observado, que a complexidade do problema aumenta em concordância ao aumento de ρ e τ.

Foram propostos dois algoritmos para a solução do problema de ADT que seguem abordagens distintas. No Capítulo 3 foi proposto o algoritmo ADTL, que utiliza uma abordagem local da alocação de tarefa do robô para atualizar sua alocação. O algoritmo é executado em cada robô de forma descentralizada realizando a atualização da alocação de tarefa do robô a partir de uma avaliação determinística do conhecimento atual que este possui sobre as tarefas alocadas aos demais robôs do enxame.

No Capítulo 4, foi proposto o algoritmo ADTG que utiliza a abordagem global da alocação de tarefas do enxame para atualizar a alocação do enxame com base no algoritmo de otimização bio-inspirado *PSO*. Cada robô possui uma possível solução para a alocação do enxame que é continuamente atualizada através da troca de mensagens entre os robôs. As alocações são avaliadas quanto a sua aptidão em ser uma alocação que atenda à proporção objetivo. Quando é identificada a alocação de maior aptidão no enxame, todos os robôs do enxame são alocados para as tarefas definidas por esta alocação.

Os algoritmos propostos foram implementados em enxames de robôs reais do tipo Elisa III, para a realização dos ensaios, conforme mostrado nos Capítulo 5. Um protocolo de comunicação otimizado foi implementado durante a troca de mensagens entre os robôs. A implementação do protocolo otimizado permitiu melhorar o rendimento da comunicação, tornado-a quatro vezes mais rápida.

Uma análise dos resultados obtidos para os ensaios realizados em enxames de diferentes arranjos (ρ e τ) foi apresentada no Capítulo 6. Os resultados avaliaram o desempenho dos algoritmos quanto ao aspecto de convergência e comunicação. No aspecto de convergência foi analisado o tempo de convergência do algoritmo e no aspecto de comunicação o número de mensagens trocadas entre os robôs até que a proporção-objetivo fosse alcançada.

Nos aspectos de convergência, todos os resultados apresentados demonstraram a eficácia de ambos os algoritmos em convergir para uma nova alocação que atendesse a proporção objetivo. Entretanto, foi observada em alguns ensaios para o algoritmo ADTL uma instabilidade na ação de manter a alocação de tarefas após o enxame atingir a

convergência, de modo que o algoritmo tivesse que intervir novamente para altera-la. Observou-se a superioridade do algoritmo ADTL em relação ao algoritmo ADTG na questão de tempo de convergência, uma vez que ADTL alcançou a proporção objetivo em menor tempo que ADTG em todos os ensaios realizados.

Quanto ao aspecto de comunicação, ADTL mostrou necessitar de um número menor de mensagens trocadas para que o enxame atingisse a proporção objetivo, quando comparado à ADTG. Esta característica é decorrente da implementação dos algoritmos propostos, de modo que o algoritmo ADTL necessita de um número menor de mensagens para convergir para a proporção objetivo quando comparado à ADTG. Devido a isto, o tempo de convergência é superior no algoritmo ADTG em relação ao algoritmo ADTL.

7.2 Trabalhos Futuros

Nesta seção, são citadas algumas possíveis melhorias nos algoritmos propostos com o intuito de melhorar o seu desempenho. Também são incentivados a realização de novos ensaios que busquem avaliar diferentes aspectos dos algoritmos propostos.

Como sugestão de nova avaliação aos algoritmos propostos é incentivada a realização de ensaios em que sejam avaliados os aspectos de robustez de cada algoritmo. A partir da simulação de ocorrência de falhas, tais como perda momentânea da comunicação e retirada ou inclusão de robôs no enxame, seriam avaliadas a capacidade de cada algoritmo em adequar-se as novas características do problema.

Por fim, um possível acréscimo neste tema seria uma nova proposta de algoritmo de alocação de tarefas que realizasse a troca de mensagens limitando-se a vizinhança de cada robô. Nesta nova proposta, cada robô do enxame trocaria mensagens unicamente com um número restrito de robôs. Desta forma, a atualização de seu conhecimento a respeito da alocação dos demais seria ainda mais limitada exigindo do algoritmo um processo de decisão mais sofisticado para realizara alocação de tarefa ao robô.

Referências Bibliográficas

Kennedy, J. e R. Eberhart: Particle Swarm Optimization: In: *IEEE International Conference on Neural Network*, pp. 1942-1948. IEEE Press, Australia (1995)

Engelbrecht, Andrews P: Fundamentals of Computational Swarm Intelligence. John Wiley & Sons Ltd., New Jersey (2005)

Nedjah, N., Coelho, L.S. and de Macedo Mourelle, L.: Multi-Objective Swarm Intelligent Systems − Theory & Experiences. Springer, Berlin (2010)

Tewolde, G.S., Hanna, D.M. and Haskell, R.E.: 'Accelerating the performance of particle swarm optimization for embedded applications': *Congress on Evolutionary Computation*, 2009.

Muñoz, D.M., Llanos, C.H., Coelho, L.S., Mauricio Ayala-Rincín: 'Hardware architecture for PSO using floating-point arithmetic': *Ninth International Conference on Intelligent Systems Design and Applications*, pp. 243-248, 2009.

Li, S-A., Wong, C-C., Yu, C-J. and Hsu, C-C.: 'Hardware/Software Co-design for Particle Swarm Algorithm', *IEEE International Conference on Systems, Man, and Cybernetics*,pp. 3762-3767, 2010.

Nedjah, N., Coelho, L.S. and de Macedo Mourelle, L.: 'Multi-Objective Swarm Intelligent Systems − Theory & Experiences', Vol 261., Springer, Berlin, 2010.

Maeda, Y. and Matsushita, N.:'Simultaneous perturbation particle swarm optimization using FPGA', *International Joint Conference on Neural Networks*, pp. 2695 -2700, 2007.

Shutte, J.F., Reinbolt, J.A., Fregly, B.J., Haftka, R.T. and George, A.D.: 'Parallel global optimization with the particle swarm algorithm', *NIH Public Access, Int J. Numer. Methods Eng.*, Vol. 61, N. 13, Dec 2004.

Koh, B., George, A.D., Haftka, R.T. and Fregly, B.J.: 'Parallel asynchronous particle swarm algorithm', *NIH Public Access, Int J. Numer. Methods Eng.*, Vol. 67, No. 4, pp. 578-595, Jul 2006.

Calazan, R..M, Nedjah, N., and de Macedo Mourele, L.: Parallel co-processor for PSO, Int. J. High Performance Systems Architecture, 3:4, (2011)

Calazan, R.M., Nedjah, N., and de Macedo Mourele, L.: A Massively Parallel Reconfigurable Co-processor for Computationally Demanding Particle Swarm Optimization, In: *3rd. International Symposium of IEEE Circuits and Systems in Latin America — LASCAS'2012*, IEEE Computer Press, CA:Los Alamitos (2012)

NVIDIA: NVIDIA CUDA C Programming Guide, Version 4.0 NVIDA Corporation(2011)

NVIDIA: CURAND Library, Version 1.0, NVIDA Corporation(2010)

David B. Kirk, Wen-mei W. Hwu: Programming Massively Parallel Processors. Morgan Kaufmann, San Francisco (2010)

Jason Sanders, Eduard Kandrot: CUDA by Example, An Introduction to General-Purpose GPU Programing. Addison-Wesley, San Francisco (2010)

Veronese, L., Krohling, R. A.: Swarm's flight: accelerating the particles using C-CUDA. In: 11th IEEE Congress on Evolutionary Computation, pp. 3264-3270. IEEE Press, Trondheim (2009).

Zhou, Y., Tan, Y: GPU-based parallel particle swarm optimization. In: 11th IEEE Congress on Evolutionary Computation (CEC 2009), pp. 1493-1500. IEEE Press, Trondheim (2009).

Cádenas-Montes, M., Vega-Rodríguez, M.A., Rodríguez-Vázquez, J.J., Gómez-Iglesias, A.,: Accelerating Particle Swarm Algorithm with GPGPU. In: 19th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), pp. 560-564. IEEE Press, Cyprus (2011).

Weihang Zhu, Curry, J.: Particle Swarm with graphics hardware acceleration and local pattern search on bound constrained problems. In: IEEE Swarm Intelligence Symposium, SIS '09. pp. 1-8. IEEE Press, Nashville (2009).

REFERÊNCIAS

ALAMI, R. et al. Multi-robot cooperation in the martha project. *Robotics & Automation Magazine, IEEE*, IEEE, v. 5, n. 1, p. 36–47, 1998.

ALUR, R. et al. The algorithmic analysis of hybrid systems. *Theoretical computer science*, Elsevier, v. 138, n. 1, p. 3–34, 1995.

ARKIN, R. C.; BALCH, T.; NITZ, E. Communication of behavorial state in multi-agent retrieval tasks. In: IEEE. *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*. [S.l.], 1993. p. 588–594.

BAGHAEI, K. R.; AGAH, A. *Task allocation methodologies for multi-robot systems*. [S.l.], 2002.

BALCH, T.; PARKER, L. E. *Robot teams: from diversity to polymorphism*. [S.l.]: AK Peters Wellesley, 2002.

BAYINDIR, L.; SAHIN, E. A review of studies in swarm robotics. *Turkish Journal of Electrical Engineering*, v. 15, n. 2, p. 115–147, 2007.

BOTELHO, S. C.; ALAMI, R. M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement. In: IEEE. *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*. [S.l.], 1999. v. 2, p. 1234–1239.

CAO, Y. U. et al. Cooperative mobile robotics: Antecedents and directions. In: IEEE. *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*. [S.l.], 1995. v. 1, p. 226–234.

CASTELPIETRA, C. et al. Coordination in multi-agent autonomous cognitive robotic systems. In: CITESEER. *Proceedings of 2nd International Cognitive Robotics Workshop*. [S.l.], 2000.

CHAIMOWICZ, L.; CAMPOS, M. F.; KUMAR, V. Dynamic role assignment for cooperative robots. In: IEEE. *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*. [S.l.], 2002. v. 1, p. 293–298.

COLORNI, A. et al. Distributed optimization by ant colonies. In: PARIS, FRANCE. *Proceedings of the first European conference on artificial life*. [S.l.], 1991. v. 142, p. 134–142.

DIAS, M. B.; STENTZ, A. A free market architecture for distributed control of a multirobot system. In: *6th International Conference on Intelligent Autonomous Systems (IAS-6)*. [S.l.: s.n.], 2000. p. 115–122.

DROGOUL, A.; FERBER, J. From tom thumb to the dockers: Some experiments with foraging robots. *From Animals to Animats II*, p. 451–459, 1993.

EBERHART, R.; KENNEDY, J. A new optimizer using particle swarm theory. In: IEEE. *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*. [S.l.], 1995. p. 39–43.

EMERY, R.; SIKORSKI, K.; BALCH, T. Protocols for collaboration, coordination and dynamic role assignment in a robot team. In: IEEE. *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*. [S.l.], 2002. v. 3, p. 3008–3015.

ENGELBRECHT, A. P. *Fundamentals of computational swarm intelligence*. [S.l.]: Wiley Chichester, 2005.

GE, S. S.; CUI, Y. J. New potential functions for mobile robot path planning. *Robotics and Automation, IEEE Transactions on*, IEEE, v. 16, n. 5, p. 615–620, 2000.

GERKEY, B. P.; MATARIĆ, M. J. Murdoch: Publish/subscribe task allocation for heterogeneous agents. In: ACM. *Proceedings of the fourth international conference on Autonomous agents*. [S.l.], 2000. p. 203–204.

GERKEY, B. P.; MATARIC, M. J. Sold!: Auction methods for multirobot coordination. *Robotics and Automation, IEEE Transactions on*, IEEE, v. 18, n. 5, p. 758–768, 2002.

GERKEY, B. P.; MATARIĆ, M. J. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, SAGE Publications, v. 23, n. 9, p. 939–954, 2004.

HAN, K.-H.; KIM, J.-H. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. *Evolutionary Computation, IEEE Transactions on*, IEEE, v. 6, n. 6, p. 580–593, 2002.

HO, S.-Y. et al. Opso: Orthogonal particle swarm optimization and its application to task assignment problems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, IEEE, v. 38, n. 2, p. 288–298, 2008.

IDE, A. [S.l.], 2014. Disponível em: <http://arduino.cc/en/main/software>.

IOCCHI, L.; NARDI, D.; SALERNO, M. Reactivity and deliberation: a survey on multi-robot systems. In: *Balancing reactivity and social deliberation in multi-agent systems*. [S.l.]: Springer, 2001. p. 9–32.

JR, F. S.; THOMAS, G. Directed stigmergy-based control for multi-robot systems. In: ACM. *Proceedings of the ACM/IEEE international conference on Human-robot interaction*. [S.l.], 2007. p. 223–230.

JUNG, D.; ZELINSKY, A. Grounded symbolic communication between heterogeneous cooperating robots. *Autonomous Robots*, Springer, v. 8, n. 3, p. 269–292, 2000.

KALRA, N.; MARTINOLI, A. Comparative study of market-based and threshold-based task allocation. In: *Distributed Autonomous Robotic Systems 7*. [S.l.]: Springer, 2006. p. 91–101.

KENNKDY, J.; KBEHHART, R. Particle swarm optimization!; c. In: *Proc of IEEE International Conference on Neural Network. Piscataway: IEEE Press*. [S.l.: s.n.], 1995. p. 1942–1948.

LIU, S.; ZHANG, Y. Multi-robot task allocation based on particle swarm and ant colony optimal. *Journal of Northeast Normal University*, v. 41, n. 4, p. 68–72, 2009.

LIU, S. et al. Multi-robot task allocation based on swarm intelligence. *Journal of Jilin University*, v. 40, n. 1, p. 123–129, 2010.

LYNCH, N. A. *Distributed algorithms*. [S.l.]: Morgan Kaufmann, 1996.

MATARIC, M. J. Designing emergent behaviors: From local interactions to collective intelligence. In: *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*. [S.l.: s.n.], 1993. p. 432–441.

MENDONÇA, R. M. de; NEDJAH, N.; MOURELLE, L. de M. Swarm robots with queue organization using infrared communication. In: *Computational Science and Its Applications–ICCSA 2012*. [S.l.]: Springer, 2012. p. 136–147.

MENDONÇA, R. M. de; NEDJAH, N.; MOURELLE, L. de M. Efficient distributed algorithm of dynamic task assignment for swarm robotics. In: *Computational Science and Its Applications–ICCSA 2013*. [S.l.]: Springer, 2013. p. 500–510.

MOHAN, Y.; PONNAMBALAM, S. An extensive review of research in swarm robotics. In: IEEE. *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. [S.l.], 2009. p. 140–145.

MURPHY, R. *An introduction to AI robotics*. [S.l.]: The MIT press, 2000.

NORDIC. *nRF24L01+ Single Chip 2.4GHz*. [S.l.], 2008. Disponível em: <http://www.nordicsemi.com/kor/Products/2.4GHz-RF/>.

PARKER, L. E. *Heterogeneous multi-robot cooperation*. [S.l.], 1994.

PARKER, L. E. Alliance: An architecture for fault tolerant multirobot cooperation. *Robotics and Automation, IEEE Transactions on*, IEEE, v. 14, n. 2, p. 220–240, 1998.

PARKER, L. E. Current research in multirobot systems. *Artificial Life and Robotics*, Springer, v. 7, n. 1-2, p. 1–5, 2003.

PUGH, J.; MARTINOLI, A. Multi-robot learning with particle swarm optimization. In: ACM. *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. [S.l.], 2006. p. 441–448.

REYNOLDS, C. W. Flocks, herds and schools: A distributed behavioral model. In: ACM. *ACM SIGGRAPH Computer Graphics*. [S.l.], 1987. v. 21, n. 4, p. 25–34.

RUS, D.; DONALD, B.; JENNINGS, J. Moving furniture with teams of autonomous robots. In: IEEE. *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*. [S.l.], 1995. v. 1, p. 235–242.

RYLAND, G. G.; CHENG, H. H. Design of imobot, an intelligent reconfigurable mobile robot with novel locomotion. In: IEEE. *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. [S.l.], 2010. p. 60–65.

SAHIN, E. et al. Swarm-bot: Pattern formation in a swarm of self-assembling mobile robots. In: IEEE. *Systems, Man and Cybernetics, 2002 IEEE International Conference on*. [S.l.], 2002. v. 4, p. 6–pp.

SALMAN, A.; AHMAD, I.; AL-MADANI, S. Particle swarm optimization for task assignment problem. *Microprocessors and Microsystems*, Elsevier, v. 26, n. 8, p. 363–371, 2002.

SARIEL, S. *An integrated planning, scheduling and execution framework for multi-robot cooperation and coordination*. Tese (Doutorado) — Phd thesis, Istanbul Technical University, Turkey, 2007.

SARIEL, S.; BALCH, T. A distributed multi-robot cooperation framework for real time task achievement. In: *Distributed Autonomous Robotic Systems 7*. [S.l.]: Springer, 2006. p. 187–196.

SARIEL, S.; BALCH, T.; ERDOGAN, N. Naval mine countermeasure missions. *Robotics & Automation Magazine, IEEE*, IEEE, v. 15, n. 1, p. 45–52, 2008.

SHOR, P. W. Algorithms for quantum computation: discrete logarithms and factoring. In: IEEE. *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*. [S.l.], 1994. p. 124–134.

SMITH, R. G. The contract net protocol: High-level communication and control in a distributed problem solver. *Computers, IEEE Transactions on*, IEEE, v. 100, n. 12, p. 1104–1113, 1980.

STILWELL, D. J.; BAY, J. S. Toward the development of a material transport system using swarms of ant-like robots. In: IEEE. *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*. [S.l.], 1993. p. 766–771.

STONE, P.; VELOSO, M. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. *Artificial Intelligence*, Elsevier, v. 110, n. 2, p. 241–273, 1999.

STROM, R. et al. Gryphon: An information flow based approach to message brokering. *arXiv preprint cs/9810019*, 1998.

TANG, F.; PARKER, L. E. Asymtre: Automated synthesis of multi-robot task solutions through software reconfiguration. In: IEEE. *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. [S.l.], 2005. p. 1501–1508.

TANG, F.; PARKER, L. E. Coalescent multi-robot teaming through asymtre: a formal analysis. In: IEEE. *Advanced Robotics, 2005. ICAR'05. Proceedings., 12th International Conference on*. [S.l.], 2005. p. 817–824.

THRUN, S. Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, Morgan Kaufmann San Mateo, CA, p. 1–35, 2002.

WEIGEL, T. et al. Cs freiburg: Doing the right thing in a group. In: *RoboCup 2000: Robot Soccer World Cup IV*. [S.l.]: Springer, 2001. p. 52–63.

WERGER, B. B.; MATARIC, M. J. Broadcast of local eligibility: behavior-based control for strongly cooperative robot teams. In: ACM. *Proceedings of the fourth international conference on Autonomous agents*. [S.l.], 2000. p. 21–22.

YANG, D.; WANG, Z.-o. Improved ant algorithm for assignment problem [j]. *Journal of Tianjin University*, v. 4, p. 020, 2004.

YINGYING, D.; YAN, H.; JINGPING, J. Multi-robot cooperation method based on the ant algorithm. In: IEEE. *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*. [S.l.], 2003. p. 14–18.

YU, Z. et al. A quantum-inspired ant colony optimization for robot coalition formation. In: IEEE. *Control and Decision Conference, 2009. CCDC'09. Chinese*. [S.l.], 2009. p. 626–631.

YUPING, W.; YINGHUA, L. A novel quantum genetic algorithm for tsp. *Chinese journal of computers*, v. 30, n. 5, p. 748–755, 2007.

ZHANG, Y. et al. Large-scale multi-robot task allocation based on ant colony algorithm. In: IEEE. *Control and Decision Conference, 2008. CCDC 2008. Chinese*. [S.l.], 2008. p. 2141–2146.

ZHANG, Y.; LIU, S.-H. Survey of multi-robot task allocation. *CAAI Transactions on Intelligent Systems*, v. 3, n. 2, p. 115–120, 2008.

ZLOT, R. et al. Multi-robot exploration controlled by a market economy. In: IEEE. *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*. [S.l.], 2002. v. 3, p. 3016–3023.