

Proposta de uma Arquitetura para Alocação de Tarefas em Grupos de Robôs Móveis Baseada em Acordo Bizantino

Telmo dos Santos Klipp
Grupo de Automação e Robótica Inteligente
Universidade Federal do Rio Grande
Rio Grande, Brasil
tsklipp@furg.br

Anderson Luiz Fernandes Perez
Laboratório de Automação e Robótica Móvel
Universidade Federal de Santa Catarina
Araranguá, Brasil
anderson.perez@ufsc.br

Abstract—Mobile robots are increasingly being used to perform many tasks, such as in industry, service or military applications. Many tasks are complex and require the adoption of more than one robot to complete its execution. Thus, the formation of groups of robots is critical and must be based on the characteristics of the task to be performed, that is, the functional requirements it takes to carry it out, but also ensuring that there is a failure of one or more robots, the task can be completed, even in a longer computationally time. Thus, the robot group members should be chosen based on physical criteria, behavioral (skills) and fault tolerance. One way to ensure the execution of a task by a group of robots is to adopt some fault tolerance technique, such as byzantine agreement. Byzantine Agreement is based on the principle that existing failure of one robot in the group there will be many others who will continue the task. This paper describes a proposed architecture for task allocation in multirobot system based on MRTA taxonomy and byzantine agreement to ensure consensus and resilience among members of a group of robots.

Keywords: *mobile robotics, multirobot systems, byzantine agreement, task allocation, election algorithm.*

I. INTRODUÇÃO

Ao longo dos tempos a robótica móvel vem sendo aplicada a uma gama de tarefas [1], tais como: exploração espacial (JPL - NASA), sondas de inspeção de dutos de petróleo e gás (robô G.I.R.I.N.O - Petrobrás), cirurgias (*Da Vinci Surgical System - Intuitive Surgical*), eletrodomésticos (robô *Roomba - iRobot*), entre outras. Em algumas aplicações o uso de apenas um único robô não é o suficiente para a solução do problema, neste caso faz-se necessário à utilização de um grupo de robôs móveis.

Existem problemas que podem ser complexos o suficiente para exigir uma divisão em partes e distribuição entre entidades especializadas nessas partes para uma possível solução. Nessa divisão de tarefas, vê-se que, o termo distribuído permite uma série de benefícios no que tange a utilização de recursos, podendo-se aplicar vários conceitos computacionais que otimizam esse processo, como, paralelismo, multiprocessamento, divisão da carga de trabalho, entre outros.

O uso de robôs especializados no provimento de serviços é cada vez mais corrente em vários setores de atividades humanas, é uma tendência atual e futura, que se configura em atividades que são tediosas, repetitivas, insalubres, de risco, no entanto, alguns desses serviços não podem ser realizados por um único robô, sendo necessária a utilização de um grupo destes [2].

O uso e o compartilhamento de tarefas entre robôs móveis pode significar uma vantagem na execução destas. Segundo [3], tarefas inerentemente distribuídas (espaço, tempo, funcionalidade), como busca e resgate, exploração, e combate a incêndios são adequadas para o uso de um grupo de robôs.

Para que seja possível o compartilhamento de tarefas entre um grupo de robôs móveis é necessário a adoção de algum critério de alocação. A alocação de tarefas deve considerar que esta pode ser executada por um único robô, ou por mais de um robô, e ainda, que alguns robôs possam realizar mais de uma tarefa [4].

Em Sistemas com Múltiplos Robôs (SMR's) a comunicação entre os robôs é essencial e deve incorporar confiabilidade e segurança na troca de mensagens, para que um grupo de robôs possa entrar em concordância sobre condições aplicadas ao mesmo (aceitação de uma tarefa). Os robôs pertencentes ao grupo, além de manter o conhecimento sobre outros membros, devem ter um meio para provar a origem das tarefas que passam a receber, pois existem inúmeras possibilidades de corrupção das atividades desses robôs, seja por códigos maliciosos ou mecanismos de ataque externos, por interesses individuais de um robô, por um robô do grupo ficar inoperante, entre outros fatores.

Em razão disso será proposta uma abordagem confiável através de acordo bizantino, para garantir que a comunicação entre o grupo seja tolerante a falhas eventuais, que não podem ser previstas.

Este artigo está organizado como segue: na Seção II as formas de compartilhamento de tarefas em sistemas com múltiplos robôs; a Seção III descreve os mecanismos de acordo em sistemas com falhas; a arquitetura proposta, intitulada MRTA-B, é descrita na Seção IV; uma avaliação

baseada em estudo de caso da arquitetura MRTA-B é apresentada na Seção V; e as considerações finais são apresentadas na Seção VI.

II. COMPARTILHAMENTO DE TAREFAS EM ROBÓTICA MÓVEL

A. Tarefas em Robótica Móvel

Uma forma de descrever uma tarefa em robótica móvel é defini-la como uma atividade a ser iniciada com requisitos mínimos e um objetivo específico, que pode ser realizada por uma ou mais entidades. Os requisitos mínimos de uma tarefa podem ser descritos como habilidades necessárias a sua execução, tal como, a detenção de uma ou mais ferramentas específicas, a capacidade de atender a um limite de tempo, um estado específico da entidade executora, etc.

Algumas habilidades básicas de robôs móveis são mapeamento, navegação e localização [5]. Estas capacidades são largamente estudadas em robótica móvel sobre diferentes abordagens e técnicas. A navegação condiz com a capacidade de um robô de se locomover em um ambiente seja ele fechado (*indoor* – salas, prédios, etc.) ou aberto (*outdoor* – campos, oceano, cidade, etc.). A localização é a descoberta pelo robô de sua posição no ambiente, seja local (parte do ambiente visível pelo robô) ou global (ambiente total em que o robô está imerso). Já o mapeamento refere-se a construção de mapas do ambiente e sua relação com o que o robô percebe a sua volta, um mapa pode ser conhecido a priori ou construído pelo robô a medida que navega e visualiza o ambiente. Para uma abordagem mais completa, essas três habilidades são combinadas.

Além das três habilidades básicas citadas pode ser adicionado a robôs uma grande variação de capacidades para o cumprimento de atividades específicas. Robôs podem ter a capacidade de reconhecer objetos, manipular objetos (carregar, arrastar, agarrar, arremessar, absorver, etc.), subir escadas, voar, aquaplanar, mergulhar, perfurar materiais. Enfim, a uma enorme variação devido aos componentes computacionais, as ferramentas (sensores e atuadores), o sistema de controle, e os propósitos e problemas que o robô deve resolver.

Segundo [6], há uma interdependência entre robô, tarefa e ambiente. A tarefa de pegar um objeto situado a certa distância necessitará de um robô as capacidades de locomoção, reconhecimento e manipulação do objeto. A tarefa poderá ser dividida na sequência, localizar o objeto, se locomover até o objeto e pegá-lo. Se um robô não possuir essas três habilidades ele poderá solicitar auxílio para outros robôs que contemplam esses requisitos, isso pode ser entendido como um caso em que as capacidades estão distribuídas entre mais de um robô, mas nenhum possui as três ao mesmo tempo, sendo necessário cooperarem de alguma forma para a realização da tarefa.

Existem capacidades que permitem a realização de uma enormidade de tarefas, assim como existem tarefas que só são realizáveis por mais de uma ação, como, por exemplo, pegar um objeto a certa distância ou arrastar uma caixa por uma determinada distância. Para o problema da caixa seria necessário localizar e reconhecer a caixa, ir até a caixa, e

arrastar a caixa por uma distância determinada. A simples tarefa de reconhecer e localizar objetos em um ambiente compreende mais de uma ação. A tarefa do robô e o ambiente são determinantes para solução de um problema.

Robôs geralmente são construídos com propósitos específicos, sendo praticamente impossível construir um robô de propósito geral. Robôs móveis vêm sendo construídos para incluírem e superarem níveis de habilidades que os predisponham a realização de mais tarefas, apesar disso as tecnologias disponíveis e os custos ainda remetem os robôs a tarefas específicas e bem definidas. Algumas tarefas, porém, são mais complexas e requerem múltiplas habilidades e ações, sendo impossível sua execução por um único robô.

A utilização de SMR incide sobre tarefas complexas que necessitam o uso de mais de um robô. Nesses sistemas o principal desafio é gerenciar o uso dos diversos recursos disponíveis através do grupo de robôs, que devem se comunicar e coordenar suas ações para a execução de uma tarefa [7].

Tarefas complexas podem ser decompostas em tarefas menores e independentes sempre que possível, desta forma elas podem ser distribuídas para um grupo de robôs visando melhor aproveitamento dos recursos disponíveis. O projeto *Swarmanoid* [8] é um exemplo de aplicação de sistemas com múltiplos robôs.

B. Utilidade na Execução de Tarefas

A utilidade na realização de uma tarefa qualquer é um valor que quantifica a predisposição para sua execução. A utilidade pode ser calculada através de modelos de custos, grandezas que mensuram propriedades na execução de uma tarefa, tais como, o custo na utilização de recursos, o *fitness* (aptidão para a execução da tarefa), uma recompensa para o desempenho de um robô, ou prioridade de execução da tarefa [9].

O valor de utilidade pode surgir da combinação desses modelos de custos, comumente listados na forma de [9]:

- $Utilidade = Recompensa - Custo$
- $Utilidade = Fitness - Custo$

Em sistemas de múltiplos robôs a medição de utilidade pode ser usada para otimizar a realização de tarefas e consequentemente o desempenho do sistema. Conceitos comuns de utilidade propõem um valor único não negativo atribuído a um robô na execução de uma tarefa específica [4].

Sob o ponto de vista da otimização a mensuração de utilidade tem por objetivo o alcance do melhor desempenho para um sistema fazendo a correta relação entre as capacidades e habilidades dos robôs, as condições e estados do robô e ambiente, os requisitos das tarefas e regras de alocação de tarefas impostas para o sistema.

C. Tratamento de Tarefas

Segundo [10], a forma mais comum entre as abordagens de atribuição de tarefas para robôs é delegar uma lista com

tarefas em sua forma primitiva, ou seja, tarefas simples. Porém em sistemas de múltiplos robôs é mais natural o uso do termo missão, que é uma descrição de alto nível para uma tarefa complexa, que pode ser decomposta em sub-tarefas.

O tratamento de tarefas é realizado na forma de um planejamento sobre a questão do que deve ser feito. Em [2] é proposta uma terminologia para classificar os diferentes tipos de tarefas.

Segundo [7], a diferenciação entre os tipos de tarefas é necessária, pois no desenvolvimento de um algoritmo de alocação, tarefas complexas requerem uma modelagem de abstração em vários níveis, isso ocorre pois uma tarefa atômica não pode ser delegada a múltiplos robôs e uma tarefa composta formada por uma única decomposição é predeterminada e não alterável, sem a flexibilidade requerida por uma tarefa complexa.

Um algoritmo de alocação para tarefas complexas deve lidar com atribuições dinâmicas para produzir resultados de execução melhores, nele a decomposição (planejamento) de tarefas deve ser flexível, pois alterações podem ser necessárias [2]. Isso ocorre devido a dinamicidade no ambiente dessas tarefas, a quantidade de caminhos para soluções, as capacidades dos robôs e a quantidade de atribuições entre robôs e tarefas. Soluções melhores não são predeterminadas, mas ocorrem dinamicamente em resposta a demanda do sistema.

D. Métodos de Alocação de Tarefas entre Múltiplos Robôs

Sistemas de múltiplos robôs geralmente envolvem soluções para tarefas de caráter complexo e nível exponencial caracterizado como NP-difícil [2]. São sistemas que envolvem uma multiplicidade de formas de atribuição entre robôs e tarefas. O problema de alocação de tarefas refere-se a determinação de qual robô deve executar qual tarefa para alcançar um objetivo global [4].

A alocação é necessária para tarefas que exijam um grupo de robôs detentor de recursos limitados e dispersos entre seus membros. A alocação pode ser descrita como um método que melhor faça uso desses recursos alcançando a execução da tarefa, levando em consideração a interação dos membros do grupo e as alterações no ambiente.

No de trabalho de [4], é proposta uma taxonomia para classificar problemas comuns de MRTA facilitando pesquisas e trabalhos na área de SMR. As seguintes definições são dadas:

- Robôs de única tarefa (ST - *Single-task robots*) vs Robôs de múltiplas tarefas (MT - *Multi-task robots*):
- ST: cada robô pode executar apenas uma tarefa em um intervalo de tempo.
- MT: cada robô pode executar mais de uma tarefa no mesmo intervalo de tempo.
- Tarefas de único robô (SR - *Single-Robot tasks*) vs Tarefas de múltiplos robôs (MR - *Multi-Robot tasks*):

- SR: cada tarefa requer apenas um robô para ser completada
- MR: cada tarefa requer mais de um robô para ser completada.
- Atribuição instantânea (IA - *Instantaneous Assignment*) vs Atribuição de tempo estendido (TA - *Time-Extended Assignment*):
- IA: a informação disponível sobre os robôs, as tarefas, e o ambiente somente permite a alocação instantânea de tarefas para cada robô, não há planejamento para futuras alocações.
- TA: existe mais disponibilidade de informação, como conhecimento de todas as tarefas que precisam ser atribuídas, ou um modelo de como às tarefas iram chegar ao longo do tempo. Há a possibilidade de um planejamento sobre alocações de tarefas.

A combinação dos itens acima descritos forma uma taxonomia onde os problemas de MRTA podem ser incluídos permitindo a classificação de seus domínios, facilitando assim, o desenvolvimento de uma abordagem de solução, seja a aplicação de algoritmos ou compreensão do nível de complexidade [4].

III. ACORDOS EM SISTEMAS COM FALHAS

Acordos entre processos em sistemas distribuídos é um requisito fundamental em um grande número de aplicações. Existem formas de coordenação entre processos que requerem a troca de informações para uma negociação em busca de entendimento em comum ou acordo [11].

A utilização de acordos entre processos em sistemas distribuídos pode aumentar a tolerância a falhas, ao introduzir-se a capacidade de identificação de componentes faltosos e dos componentes capazes de dar continuidade ao sistema.

Em contraste, geralmente com sistemas presentes em uma única máquina onde uma falha pode levar a uma parada total, sistemas distribuídos podem evitar danos maiores por falhas isolando e ignorando um componente faltoso [12].

Para uma melhor compreensão das falhas que tem importância em determinados sistemas são necessários modelos de falha. Existem classes de modelos de falha massivamente estudados, que especificam maneiras em que componentes de um sistema podem falhar [11], [12].

A melhor forma para tornar um sistema tolerante a falhas é utilizando redundância, que pode ser [12]:

- Redundância de informação: são adicionados bits extras com a intenção de permitir a recuperação de bits de dados deteriorados por meio de alguma técnica, a exemplo do código de *Hamming*.
- Redundância de tempo: se necessário, uma ação realizada é executada novamente.
- Redundância física: pode ser tanto de hardware como de software, sendo que são adicionados equipamentos

ou processos extras para que um sistema tolere a falha em alguns componentes.

Um objetivo ao se utilizar modelos de erros é tornar um sistema resiliente. A resiliência é descrita como a propriedade em que um objeto abstrato ou físico retorna a forma original ou posição após sofrer alterações ou perturbações. Representa a capacidade de regresso a forma original de equilíbrio após recuperação e superação contra adversidades.

Um sistema é K -tolerante a falhas se este puder sobreviver a falhas em K componentes cumprindo suas especificações, como entendido para alguns casos [12]:

- Falhas silenciosas: K componentes param sem propagar mensagens com erros para outros componentes, nesse caso é necessário $K+1$ de redundância para manter o sistema tolerante a falhas.
- Falhas bizantinas: é necessário $2K + 1$ de redundância de componentes que funcionam corretamente, para manter o sistema tolerante a falhas de um total de $3K + 1$ componentes.

Um aspecto chave para contornar falhas de processos é organizar vários processos idênticos em um grupo no qual todos os processos recebam mensagens enviadas por qualquer membro do grupo [12].

Sistemas com vários nós processadores, dentre esses os sistemas multi-robôs, podem falhar de maneira arbitrária, sendo que, uma falha é vista como um comportamento incomum ao especificado para um sistema, e é oriunda de um ou mais erros no estado do sistema [11].

A falha arbitrária (ou bizantina) e o tipo de falha mais severa e a menos restritiva, pois encapsula todos os modelos de falhas conhecidos [13].

Em um sistema com acordo bizantino uma negociação deve ser iniciada por um processo (robô), por meio de um valor inicial que deve ser acordado entre os demais membros do grupo, sempre que satisfeitas as condições descritas por [11]:

- Acordo - todo processo confiável deve concordar com

o mesmo valor.

- Validação - se o processo que inicia um acordo é confiável todos os demais processos devem concordar com o mesmo valor inicial dado pelo processo que iniciou a negociação.
- Terminação - eventualmente todos os processos não faltosos devem concordar sobre um valor.

Para garantir a correção de falhas e a manutenção das funções de um sistema distribuído onde nós comunicam-se trocando informações através de mensagens estabelecendo um acordo bizantino, alguns algoritmos podem ser utilizados, tais como: BFT (*Byzantine Fault Tolerance*), *Consensus Algorithm* (CA), *Interactive Consistency Algorithm* (ICA), *Signed Messages* (SM).

IV. ARQUITETURA DE COMPARTILHAMENTO DE TAREFAS

A. Visão Geral da Arquitetura Proposta

A arquitetura proposta é baseada na taxonomia para alocação de tarefas em sistemas com múltiplos robôs. Dessa forma, como se pretende estabelecer acordo bizantino nas comunicações entre os robôs, a arquitetura proposta foi intitulada MRTA-B.

A taxonomia MRTA visa a classificação de problemas de alocação de tarefas em sistemas de multi-robôs, sendo a sigla MT-MR-TA (*Mult task robots – Mult robot tasks – Time extended assignment*) a classificação de problemas que a arquitetura MRTA-B pretende lidar, ou seja, a alocação de tarefas complexas entre multi-robôs.

A arquitetura MRTA-B visa a alocação de tarefas entre grupos de robôs aptos a executá-las, porém fazendo uso de técnicas que possibilitem a tolerância de falhas eventuais e inesperadas no SMR, necessitando, desta forma, o estabelecimento de acordo entre as partes envolvidas na execução de uma dada tarefa. A arquitetura MRTA-B é ilustrada na Fig. 1.

Os componentes presentes na estrutura da arquitetura

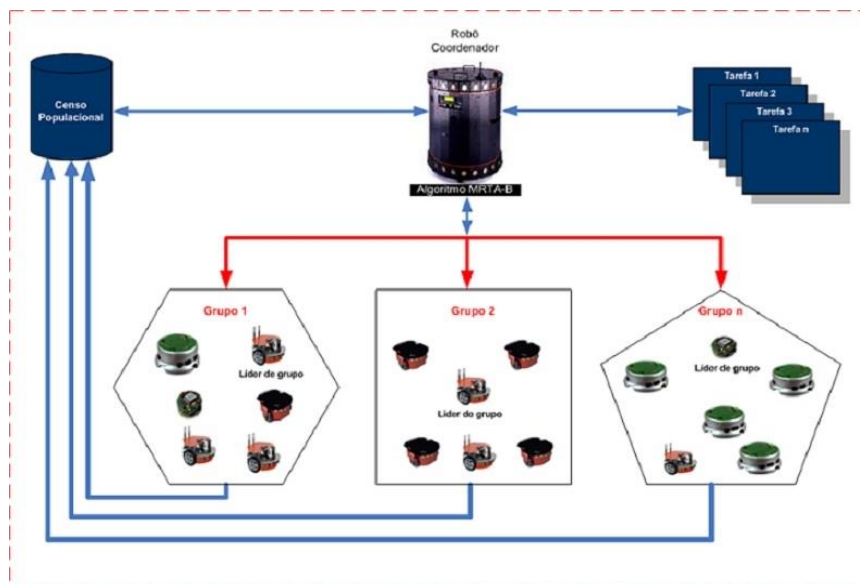


Fig. 1. Visão Geral da Arquitetura MRTA-B.

MRTA-B são condicionados pelas interações que mantêm entre si e processos pelos quais devem passar, assim é formada uma hierarquia com diferentes fases na comunicação, sendo que, todo o processo é iniciado com uma população de robôs.

Uma população será formada por vários robôs presentes em um sistema de múltiplos robôs heterogêneos. Inicialmente cada membro da população de robôs não terá qualquer vínculo. Robôs que passam a integrar a população devem anunciar suas características e habilidades na base de dados intitulada “Censo Populacional”.

O censo populacional consiste em uma base de dados com informações sobre cada membro da população de robôs. Estas informações são as características anatômicas (físicas) e as abstrações de habilidades advindas do software chamadas de capacidades.

A Fig. 2 ilustra o processo de anúncio das capacidades pelos robôs da população.

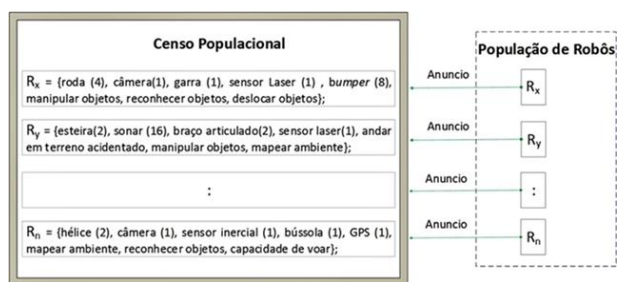


Fig. 2. Anúncio das capacidades por parte dos robôs da população.

Robôs possuem habilidades que os permitem atuar em uma ou mais atividades, sendo necessário descobrir quais robôs podem cooperar e formar um grupo. Dessa forma tarefas de transporte, vigilância patrimonial ou limpeza, devem ser atribuídas a grupos específicos de robôs.

Uma tarefa será descrita por um BDT (Bloco de Descrição da Tarefa) com os requisitos necessários que constituem uma demanda de trabalho para sua execução. A Fig. 3 ilustra um exemplo de um BDT.

Os itens que compõem um BDT são:

- Tipo da tarefa: uma pequena descrição sobre a tarefa.
- Identificação: um número que identifica a tarefa, para efeito de organização, manipulação e controle das tarefas presentes no sistema de múltiplos robôs.
- Tempo de duração: o tempo de duração de uma tarefa pode ser determinado ou indeterminado. Dessa forma uma tarefa pode ter um fim conhecido ou transcorrer infinitamente.
- Ambiente de execução: robôs podem estar aptos a trabalharem em determinados tipos de ambientes, e mesmo contendo as capacidades necessárias devem ser descartados se não atuam no ambiente exigido pela tarefa. O ambiente de execução pode ser em locais fechados (*indoor*) ou abertos (*outdoor*).



Fig. 3. Exemplo de um bloco de descrição da tarefa.

- Componentes físicos: identifica os componentes necessários que os robôs devem possuir para a realização da tarefa.
- Habilidades: as habilidades dão maior aporte sobre as reais capacidades de um robô. Estas podem definir métricas e tolerância sobre propriedades físicas, podendo estabelecer a capacidade de carga que um robô pode transportar, objetos que pode reconhecer, terrenos que pode se locomover, etc.

O BDT elenca requisitos necessários para que uma dada tarefa seja cumprida. Por exemplo, uma tarefa de transporte de carga deve definir o que e quanto deve ser carregado. No BDT não é necessário o registro de quais robôs devem realizar a tarefa, mas apenas elencar quais são os componentes físicos que esses robôs devem ter, deste modo a seleção dos robôs que farão parte de um grupo fica a cargo do coordenador.

A demanda de trabalho visa a associação dos robôs com capacidades condizentes aos requisitos da tarefa, desta forma, a tarefa tem informações suficientes para que a mesma seja atribuída a robôs com condições de executá-la. Para que isso aconteça, esses robôs devem estar corretamente organizados em um grupo.

B. Modelo de Comunicação na Arquitetura MRTA-B

Todos os robôs da arquitetura proposta devem estar aptos a se comunicarem. Desta forma os robôs devem ter canais de comunicação entre si. A comunicação entre robôs da arquitetura MRTA-B inicia-se com a necessidade de robôs apresentarem comportamentos sociais, como o da eleição de coordenador e líderes de grupo. Para que isso aconteça um robô deve ter conhecimento do destinatário de uma mensagem, e da mesma forma, reconhecer o remetente.

Como os robôs anunciam suas capacidades no censo populacional, suas identificações vão sendo indexadas. Se um robô qualquer resolve iniciar um processo de eleição, este pode criar um vetor com a identificação de cada robô que participará do processo e enviá-lo em uma mensagem a cada um dos participantes de forma síncrona. Assim, todos receberão a mensagem e poderão formar canais de comunicação.

Há duas principais atividades que podem acontecer com a população de robôs:

- Eleição de coordenador e líder – os robôs enviam mensagens em conformidade com o algoritmo do anel, ou seja, enviam somente para o primeiro sucessor ativo. A única mensagem transmitida em *broadcast* é do aviso do vencedor.
- Algoritmo do consenso – os robôs enviam as mensagens em broadcast, ou seja, para todos os robôs, a menos, que seja uma mensagem replicada, neste caso o remetente não recebe sua mensagem novamente.

As principais comunicações são:

- Entre robôs que elegem um coordenador;
- De coordenador para robôs avulsos (que formaram um grupo ou novos robôs recentemente cadastrados no censo populacional);
- De coordenador para líderes de grupo;
- De líderes de grupo para membros de grupo;
- Entre membros de grupo.

Um robô pode se utilizar de mais de uma forma de comunicação, enviando uma mensagem a um robô específico (sucessor ou não), ou para vários robôs, isso depende do nível de comportamento social que o robô estará envolvido.

C. Formação de Grupos

Um robô da população de robôs será eleito coordenador. O processo de eleição do robô coordenador exige um número mínimo de cinco robôs na população. Esse número visa respeitar o valor mínimo de robôs para que um acordo bizantino seja possível mantendo a mesma condição para a posterior formação de ao menos um grupo de robôs na população, isso ocorre devido ao fato de o robô coordenador não participar de demandas no grupo.

É necessário $2k + 1$ robôs de redundância (robôs confiáveis) para validar um acordo, ou seja, o mínimo possível são 3 (três) robôs não faltosos, pois dessa forma há critério de desempate sobre a validação de um valor. No entanto é necessário um número de $3k + 1$ robôs para que um processo de acordo seja possível, ou seja, no mínimo 4 (quatro) participantes no acordo, pois, se houver três robôs negociando e um for faltoso, os outros dois não poderão decidir sobre a validade de um valor por não haver critério de desempate.

Dessa forma, o quinto robô que anuncia suas capacidades no censo populacional inicia o processo de eleição do robô coordenador com os robôs que anteriormente já haviam se anunciado, assim, é importante que o robô que iniciou a eleição envie uma mensagem aos demais robôs contendo todas as identificações presentes no censo populacional. O coordenador eleito terá o papel de criar grupos e atribuir tarefas a estes.

Tendo conhecimento da demanda de uma tarefa o robô coordenador consultará o censo populacional para saber quais robôs atendem os requisitos necessários para a execução da tarefa. Uma mensagem será enviada para cada robô escolhido informando que este fará parte de um grupo específico. A

formação de um grupo levará em consideração um número mínimo de quatro robôs para que seja possível o estabelecimento de acordos bizantinos. Os robôs do grupo possuem habilidades suficientes para solucionar uma tarefa em questão.

O robô coordenador ficará aguardando a confirmação de resposta dos robôs escolhidos para formarem um grupo. Quando todos os robôs responderem ao coordenador, este enviará outra mensagem a todos os robôs do grupo solicitando que iniciem o processo de eleição do líder. A mensagem enviada pelo coordenador conterá um vetor com as identificações de cada robô.

O líder de um grupo de robôs terá o papel de decidir sobre a saída ou entrada de um novo robô no grupo, pelo término ou não de uma tarefa, e pela formação de equipes no grupo.

O grupo será fechado, dessa forma as mensagens somente serão trocadas internamente ao grupo e o contato externo será feito por intermédio do robô líder do grupo, desta forma, após a criação de um grupo o robô coordenador envia uma mensagem para o líder com a identificação da tarefa a ser realizada, este então negociará com os demais membros do grupo a execução da tarefa. As mensagens trocadas entre o líder e os demais robôs do grupo deverão seguir os critérios estabelecidos pelo acordo bizantino.

Cada grupo terá como característica uma “força de trabalho” que é o somatório das capacidades de todos os robôs R_i de um grupo C_g . A Equação 1 representa a força de trabalho, que deve ser suficiente para os requisitos da tarefa:

$$C_g = \sum_{i=0}^{n-1} C R_i \quad (1)$$

Dentro de um grupo de robôs o acordo bizantino será estabelecido para atestar que há robôs confiáveis o suficiente para manter a força de trabalho exigida para a uma dada tarefa.

D. Processo de Eleição do Coordenador e Líderes de Grupo

O algoritmo escolhido para o processo de eleição do robô coordenador e do líder de grupo será o algoritmo do anel (*ring algorithm*) Chang e Roberts (1979) apud [14]. No algoritmo do anel os robôs no grupo estão dispostos em uma sequência em formato de anel, onde, cada robô conhece seu antecessor e seu sucessor. Os passos descritos no algoritmo do anel são:

1. O processo de eleição é iniciado por um robô, que envia uma mensagem invocando uma eleição com sua identificação ao seu sucessor.
2. Cada robô reenvia a mensagem anexando sua identificação ao respectivo sucessor.
3. Eventualmente a mensagem retorna a um robô que anexou sua identificação. O robô checa o vencedor, que deverá ser o robô de identificação mais alta e anuncia aos demais membros do grupo.

A Fig. 5 ilustra um exemplo do processo de eleição, sendo que, no quadro a) o robô $R2$ inicia um processo de eleição; no

quadro b) todos os robôs ativos participam da eleição enviando a mensagem original com sua própria identificação; e no quadro c) o robô R2 anuncia aos demais que o robô R4 foi eleito como novo coordenador.

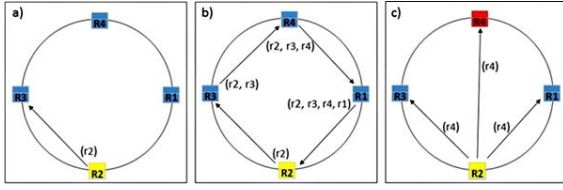


Fig. 5. Processo de eleição via algoritmo do anel.

E. Comunicação Baseada em Acordo Bizantino

As mensagens enviadas pelo líder para os membros do grupo, que podem ser de atribuição de tarefas, entrada e saída de membros no grupo, encerramento de uma tarefa, entre outras, deverão ser confiáveis.

Qualquer robô que receba uma mensagem sobre a alocação de uma tarefa deve se certificar que esta mensagem partiu do líder e se encontra íntegra, para tanto ele fará uso do acordo bizantino.

Um acordo só é possível, a partir de um grupo mínimo de quatro robôs. No entanto, para que o processo de acordo seja confiável, são necessários $2k + 1$ robôs totalmente seguros de um total de $3k + 1$ robôs membros do grupo. A equação $3f + 1$, representa o número mínimo de robôs em um grupo para que um acordo seja possível, onde f representa o número de robôs faltosos. Caso existam três robôs faltosos é necessário que um grupo seja composto de um número mínimo de 10 robôs.

É utilizado o algoritmo de Consenso (*Consensus Algorithm*), para o tratamento de falhas bizantinas especificado por [11]. Uma tarefa anunciada por um líder, será utilizada para demonstrar o funcionamento do algoritmo de consenso, e consequentemente, como é possível identificar de modo seguro se a mensagem é do líder.

Quando o líder anuncia uma tarefa aos demais robôs do grupo um processo de acordo é estabelecido. A Fig. 6 ilustra parte do processo de acordo bizantino com o algoritmo de consenso, na qual, no quadro a) o líder (destacado em vermelho) anuncia a chegada de uma tarefa ao grupo; no quadro b) o robô R2 participa de uma rodada na negociação e assim sucessivamente até o R8.

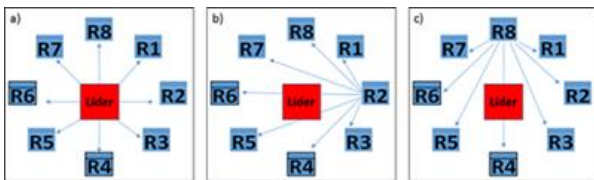


Fig. 6. Exemplo do processo de estabelecimento do acordo bizantino.

Os robôs do grupo precisam decidir sobre um valor $T(x)$ que representa a tarefa. Cada robô replica o valor $T(x)$ recebido do líder do grupo em um total de $f + 1$ rodadas, na qual f representa o número máximo de robôs faltosos que podem ser tolerados para o alcance de um acordo. Em cada rodada um robô participa do processo de negociação enviando $n - 1$ mensagens aos demais robôs no grupo. As rodadas se

repetem até o alcance de um acordo ou a não validação da tarefa por insuficiência de robôs confiáveis ou por um líder faltoso.

V. DEFINIÇÕES GERAIS

A metodologia utilizada para avaliação da arquitetura MRTA-B foi de estudo de caso, ou seja, dada uma população composta por N robôs e uma lista de tarefas é feita a análise do funcionamento da arquitetura proposta.

A arquitetura MRTA-B tem como principal característica a alocação de tarefas complexas entre grupos de robôs de forma confiável. Ela dá suporte para que robôs aptos a executarem determinada tarefa formem um grupo. Para que isso ocorra, na arquitetura existe uma hierarquia de comando, que é composta pela figura do coordenador, líder e membros de grupos. Essa hierarquia não é pré-existente, nem fixa, mas é formada dinamicamente na arquitetura MRT-B.

O algoritmo do anel e consenso garantirão, respectivamente, a eleição de coordenador e líderes de grupo e um número mínimo de robôs confiáveis para atuarem em uma tarefa. Isso implica que a arquitetura MRTA-B não leva em consideração certas condições para execução de uma tarefa, como, fracionamento e atribuição a robôs específicos ou ordem na execução de sub-tarefas, mas apenas a atribuição das tarefas supondo que os robôs tenham autonomia para executá-las.

Após anunciar suas capacidades no Censo Populacional o quinto robô inicia um processo de eleição com os outros robôs anteriormente registrados no censo visando a eleição do coordenador através do algoritmo do anel. O robô R5 envia uma mensagem requisitando uma eleição aos demais robôs. Nesse momento os robôs entram em consenso para garantir que a eleição para coordenador seja confiável, ou seja, determinando se há robôs confiáveis o suficiente para iniciar o processo de eleição do coordenador.

O processo de eleição do coordenador exige um número mínimo de cinco robôs confiáveis, pois uma vez que o coordenador é eleito, deverá haver um número mínimo de quatro robôs na população para manter futuros processos que exijam acordos bizantinos, como o da formação de um grupo.

Uma vez eleito o coordenador avisará todos os demais robôs no grupo da sua condição como coordenador, enviando uma mensagem a cada robô com o vetor de identificação dos robôs registrados no censo. Os robôs da população entrarão em processo de consenso para confirmar a veracidade da mensagem, atestando que o robô que enviou a mensagem informando ser o coordenador realmente é o coordenador.

Supondo que o robô R5 tenha sido eleito coordenador, este poderá consultar a lista de tarefas e ter acesso ao censo populacional. O robô R5 analisará a primeira tarefa, T1, e iniciará o processo de seleção dos robôs que irão formar o grupo para a execução desta tarefa. O processo de seleção dos robôs é baseado na força de trabalho, ou seja, cada robô que apresentar uma capacidade (componente físico ou habilidade) correspondente com o requisito da tarefa receberá 1 (um) ponto.

O processo de seleção e do cálculo da força de trabalho deve obedecer os seguintes critérios:

- Ambiente de execução: robôs que não atuam no ambiente exigido pela tarefa são desconsiderados.
- Habilidades: cada habilidade presente em um robô selecionado deve corresponder com a habilidade requerida pela tarefa.
- Capacidades físicas: os componentes de hardware presentes em cada robô que tenham correspondência com as exigências da tarefa.

A capacidade de ambiente de execução pode restringir de início um robô para determinada tarefa. O robô coordenador procurará inicialmente atribuir um total de 4 (quatro) robôs a partir da pontuação para a habilidade, devendo alocar ao menos um robô para cada habilidade presente como requisito na tarefa. Atendendo cada requisito de habilidade da tarefa, mas tendo um grupo incompleto, o robô coordenador passará para a pontuação dos componentes físicos, se não houver na população robôs suficientes para alocar a tarefa, o robô coordenador prosseguirá para a próxima tarefa da lista.

A eleição dos líderes ocorre internamente aos grupos após o robô coordenador enviar mensagens requisitando para que isso ocorra. Supondo, um Grupo1 de robôs, onde um robô de identificação R11, seja eleito o líder, este mandará uma mensagem para o robô coordenador R5. A partir desse momento as mensagens trocadas entre Grupo1 e o coordenador somente acontecerão via robô líder.

O coordenador R5 enviará uma mensagem para o líder R11 para que a tarefa T1 seja iniciada. O líder enviará a indicação para execução da tarefa a todos os robôs do Grupo1, que iniciarão um processo de consenso para confirmar que há robôs confiáveis o suficiente para executar a tarefa T1, sendo que se este processo for validado caberá aos robôs no grupo garantir a sua execução.

É importante ressaltar que a arquitetura proposta não leva em consideração a forma com que o robô líder irá gerenciar o processo de execução da tarefa pelos robôs liderados, aqueles pertencentes ao grupo do líder. A arquitetura proposta baseia-se no princípio de que tarefas precisam ser executadas por um grupo de robôs, logo a finalidade é criar os grupos e posteriormente fazer a alocação das tarefas a estes grupos.

VI. CONSIDERAÇÕES FINAIS

Este trabalho apresentou uma proposta de arquitetura para alocação de tarefas em sistemas multi-robôs baseada na taxonomia MRTA. A arquitetura proposta, intitulada MRTA-B, visa a formação de grupos de robôs e a alocação de tarefas de maneira confiável, isto é, garantindo que mesmo na presença de falhas arbitrárias por parte de algum robô do grupo a tarefa consiga ser executada.

A arquitetura MRTA-B contempla problemas de múltiplas tarefas e múltiplos robôs (MT-MR) proposto na taxonomia MRTA. Na MRTA-B as tarefas são separadas conforme seus requisitos e associadas a grupos de robôs aptos a executá-las.

Os grupos são formados por robôs que possuem características físicas e habilidades necessárias para o cumprimento da tarefa. O uso de acordo bizantino garante que a formação de grupos e a comunicação entre o robô coordenador e os líderes de grupos e entre o líder de um grupo e os robôs membros desse grupo sejam feitas de maneira confiável.

O uso de acordo bizantino conduz a um número mínimo de 4 (quatro) robôs por grupo corroborando com os problemas aplicáveis a MT-MR em MRTA, ou seja, garantindo que uma tarefa complexa será distribuída entre vários robôs, assim, um grupo formado por n robôs pode explicitamente cooperar na execução da tarefa.

A arquitetura MRTA-B apresenta uma solução para a alocação de tarefas entre grupos de robôs, desta forma, o robô líder de cada grupo deve gerenciar e dividir a tarefa entre os robôs membros do grupo. O gerenciamento da execução da tarefa intra-grupo foge aos propósitos deste trabalho, podendo ser objeto de estudo em trabalhos futuros.

References

- [1] Brumitt, Barry Lowell. A Mission Planning System for Multiple Mobile Robots in Unknown, Unstructured, and Changing Environments. 1998. 127 f. Tese (Doutorado) Carnegie Mellon University, Pittsburgh.
- [2] Zlot, Robert Michael. An Auction-Based Approach to Complex Task Allocation for Multirobot Teams. 2006. 187 f. Tese (Doutorado) - Robotics Institute, Carnegie Mellon University, Pittsburgh.
- [3] Ribeiro, C., Costa, A., e Romero, R. (2001). Robôs Móveis Inteligentes: Princípios e Técnicas. Proc. Anais do XXI Congresso da Sociedade Brasileira de Computação -SBC-2001, pp. 258–306.
- [4] Gerkey, Brian P.; Mataric, Maja J. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. The International Journal Of Robotics Research, Thousand Oaks, p. 939-954. 6 set. 2004.
- [5] Siegwart, Roland; Nourbakhsh, Illah R.. Introduction to Autonomous Mobile Robots. Massachusetts: The Mit Press, 2004. 335 p.
- [6] Nehmzow, Ulrich. Mobile Robotics: A Practical Introduction. 2. ed. London: Springer, 2003.
- [7] CAO, Hung et al. Complex Tasks Allocation for Multi Robot Teams under Communication Constraints. In: NATIONAL CONFERENCE ON "CONTROL ARCHITECTURES OF ROBOTS", 5, 2010, Douai.
- [8] Swarmanoid.org. Swarmanoid. Disponível em: <<http://www.swarmanoid.org>>. Acesso em: 10 nov. 2012.
- [9] Mosteo, Alejandro R.; Montano, Luis. A survey of multi-robot task allocation. Zaragoza: Aragon Institute Of Engineering Research, 2010. 27 p.
- [10] Dias, M. Bernardine et al. Market-Based Multirobot Coordination: A Survey and Analysis. Pittsburgh: Carnegie Mellon Technical Report CMU-RI-TR-05-13, 2005. 25 p.
- [11] Kshemkalyani, Ajay D.; Singhal, Mukesh. Distributed Computing: Principles, Algorithms, and Systems. New York: Cambridge University Press, 2008. 736 p.
- [12] Tanenbaum, Andrew S.; Steen, Maarten Van. Sistemas Distribuídos: Princípios e Paradigmas. 2. ed. São Paulo: Person Education do Brasil, 2008. 402 p.
- [13] Luiz, Aldelir Fernando. Arquitetura para Replicação de Serviços Tolerantes a Falhas Bizantinas Baseada em Espaço de Tuplas. 2009. 131 f. Dissertação (Mestrado) - Pontifícia Universidade Católica do Paraná, Curitiba, 2009.
- [14] Coulouris, George; Dollimore, Jean; Kindberg, Tim. Sistemas Distribuídos: Conceitos e Projeto. 4. ed. Porto Alegre: Artmed Editora, 2007. 784 p.