Uso de Clones na Determinação Dinâmica do Itinerário de Agentes Móveis com Restrição Temporal

Luciana Rech, Carlos Montez and Rômulo Silva de Oliveira
Departamento de Automação e Sistemas
Programa de Pós Graduação em Engenharia Elétrica
Universidade Federal de Santa Catarina
Caixa Postal 476, CEP 88040-900.
Florianópolis, SC, Brazil.
lrech, montez, romulo@das.ufsc.br

***Abstract* – In some applications, as part of a mobile agent mission, there may be the necessity of meeting a firm end-to-end deadline, although with some itinerary flexibility due to optional resources. This paper proposes an execution model based on the concept of imprecise mobile agents with timing constraints. In this paper it is considered an execution model where agents do not previously know their itineraries. The imprecise mobile agent may choose among alternative and/or optional resources. The objective is to propose and to evaluate heuristics to be used individually or in pairs by imprecise mobile agents that must accomplish a mission within a firm deadline.**

I. INTRODUÇÃO

Dentro do contexto de mobilidade de código, uma importante área de pesquisa é a tecnologia de agentes móveis [1,2]. Um agente móvel é um elemento de software autocontido, responsável pela execução de uma tarefa, e que não está limitado ao sistema onde começa a sua execução, sendo capaz de migrar através de uma rede [2]. Agentes móveis reduzem a carga na rede, executam assíncrona e autonomamente, e podem se adaptar dinamicamente, estabelecendo assim um paradigma para a programação em ambientes distribuídos.

Um importante requisito que pode ser associado a agentes móveis é a restrição temporal. Um possível cenário para o uso de agentes móveis com restrição temporal é um sistema de geração e distribuição de energia elétrica onde os dados encontram-se fisicamente espalhados. O agente tem como função coletar dados, formar uma imagem do problema e decidir a próxima visita para coleta de dados.

Outro possível cenário é o uso de agentes móveis com restrição temporal na linha de produção de uma fábrica [3]. Neste caso, a missão do agente é supervisionar a operação da linha de produção, tendo como função enviar relatórios descrevendo a situação atual do sistema e diagnosticar falhas.

Este artigo trata de mobilidade de código em um sistema distribuído associado com restrição temporal. Em [4] foi originalmente apresentado o modelo computacional e algumas heurísticas simples que serão utilizadas neste artigo. Em [5] foi apresentada a idéia de se utilizar clones de agentes móveis imprecisos. Entretanto, não foi feita uma avaliação de desempenho apropriada apenas foi apresentado um exemplo.

Diferentemente de [5], neste artigo será apresentada uma extensa avaliação de desempenho das heurísticas para definição do itinerário. Esta avaliação de desempenho mostrará que embora heurísticas simples sejam apropriadas para intervalos de deadline específicos, a utilização de pares de clones constitui uma abordagem mais robusta para o problema.

Neste artigo, assumimos que as possibilidades de itinerário são ignoradas antes da partida do agente móvel. O desempenho das heurísticas é comparado seguindo diferentes métricas de qualidade. São descritos e analisados algoritmos para a determinação dinâmica do itinerário seguindo um diagrama de recursos específico. O objetivo deste artigo é estudar o comportamento destas heurísticas (operando individualmente ou em pares) em diferentes cenários.

Este texto está dividido em 7 seções. Na seção 2 serão discutidas algumas propostas encontradas na literatura que tratam a questão da mobilidade aliada a questão da restrição temporal. Na seção 3 o modelo computacional é brevemente descrito (a descrição completa do modelo computacional pode ser encontrada em [6]). A seção 4 contém as heurísticas utilizadas para a definição do itinerário. Na seção 5 é apresentada uma avaliação comparativa destas heurísticas. A seção 6 apresenta o método baseado em pares de agentes móveis clones. Também é apresentada uma comparação do desempenho. A seção 7 contém as conclusões.

II. TRABALHOS RELACIONADOS

Esta seção apresenta algumas abordagens que mostram agentes móveis sendo utilizados de forma a auxiliar tarefas distribuídas com restrição temporal.

No ARS [7], os agentes móveis são utilizados para reunir as informações da situação atual dos enlaces da rede, através destas informações é que são encontrados os possíveis caminhos para as tarefas que solicitam comunicação ponto a ponto em tempo real. Através da reserva de recursos e seu controle de admissão global, o ARS consegue decidir se um caminho é possível ou não. A função dos agentes é verificar a possibilidade de se utilizar um itinerário previamente definido sem afetar a qualidade da execução e cumprindo seu deadline. Os agentes móveis coletam o atual estado dos enlaces do sistema para examinar a disponibilidade de possíveis caminhos em cada nodo.

Em [8] existe uma preocupação com a quantidade de carga no agente, evitando desta forma possíveis gargalos resultantes de uma sobrecarga. Para solucionar este problema é proposta a clonagem do agente. O agente somente será clonado se não puder desenvolver todas as suas tarefas em tempo e a eficiência do agente clone e do agente original for maior que a eficiência do agente original sozinho.

Neste artigo, o uso de agentes clones é proposto para aumentar a eficiência das heurísticas de definição de itinerário, onde foi possível perceber as vantagens em se combinar – através de agentes clones – o uso das heurísticas para alcançar melhores desempenhos. A miopia do agente pode limitar a eficiência das heurísticas existentes, portanto, uma possível solução foi utilizar um agente clone que adota uma segunda heurística.

Redes neurais e técnicas de sistemas especialistas estão sendo cada vez mais encontradas em aplicações no campo de diagnóstico de ferramentas de máquinas. Em [9] os agentes móveis possuem uma importante função, eles funcionam como programas de monitoramento, o controlador central pode despachá-los para os nodos remotos onde os sinais de máquina são fornecidos de máquinas reais. A integração destas duas técnicas de inteligência artificial é particularmente apropriada para diagnóstico de falhas em máquinas em tempo real.

Algoritmos de roteamento, para definição do itinerário dos agentes móveis com restrição temporal foram propostos em [10] e [11].

Para [10] os principais fatores do desempenho no planejamento de agentes móveis são: o número de agentes móveis, o tempo consumido pelos agentes participantes e as restrições de tempo. Na tentativa de satisfazer estes requisitos, foi proposto um método de planejamento de agentes móveis com restrição de tempo para encontrar o número mínimo de agentes e o melhor itinerário. Os agentes são utilizados na recuperação de informação em um ambiente de computação distribuído, respeitando as restrições de tempo enquanto mantem-se o tempo mínimo de roteamento total. A miopia do agente não é discutida, visto que o caminho a ser percorrido já é conhecido no momento da partida do agente.

Em [11] é apresentado um algoritmo de roteamento de agentes móveis que considera o custo dos enlaces. Para auxiliar na tomada de decisão do agente, existe uma distribuição de probabilidade para o agente móvel selecionar um dos nodos vizinhos e mover-se até ele. O custo dos enlaces entre dois nodos é definido baseado nas informações de tráfego conhecidas e na distribuição de probabilidade encontrada.

Com relação a forma como é definido o itinerário, diferentemente deste artigo, em [11] vários agentes partem em busca do melhor itinerário, voltam ao nodo origem (servidor), é então selecionado o melhor itinerário, sendo que apenas um agente parte novamente para cumprir sua missão e este já conhece seu itinerário. Neste artigo, a definição do itinerário acontece de forma dinâmica.

III. EXECUTION MODEL

Nesta seção é brevemente descrito o modelo computacional apresentado em [4,5]. Este modelo descreve o comportamento de aplicações baseadas em agentes móveis imprecisos com restrição temporal, em um sistema distribuído formado por um conjunto de nodos conectados através de um serviço subjacente de comunicação.

Neste modelo de execução, cada agente móvel possui uma missão associada com a visita a um determinado grupo de nodos do sistema. Nestes nodos encontram-se os recursos necessários para o agente completar sua missão. Será assumido que os agentes não se comunicam. A missão é formada por um grupo de tarefas que devem ser executadas respeitando um deadline. Uma tarefa representa o uso de um recurso por um agente em um determinado nodo.

Um agente móvel impreciso é aquele capaz de reduzir a qualidade do resultado para conseguir atender o deadline da missão.

Neste artigo é suposto que um diagrama de recursos indica as relações de precedência e as opcionalidades existentes entre os recursos a serem utilizados pelo agente em sua missão. Como vários recursos podem aparecer replicados no sistema, é construído um diagrama de nodos que, a partir do diagrama de recursos, mostra também estas opções.

Cada recurso representa um benefício adicional para a missão do agente, cujo objetivo é maximizar o somatório dos benefícios obtidos ao longo do itinerário, respeitando as precedências. Entretanto, o agente móvel é suposto míope, ou seja, ele conhece apenas as opções imediatas do diagrama de recursos. A miopia do agente impede que o problema seja tratado através de técnicas clássicas de otimização.

Outro requisito que deve ser considerado pelo agente móvel é o deadline de cada missão, salientando a importância da escolha do melhor itinerário para cada situação apresentada. Esta seqüência de nodos visitados pelo agente móvel entre o nodo origem e o nodo destino (itinerário) é definida dinamicamente.

A. Formulação do Problema

Cada agente móvel possui uma missão $\boldsymbol{M}$ que defini a função benefício $\boldsymbol{B}$ para cada tipo de recurso $\boldsymbol{R}$. Esta função determina a quantidade de benefício $\boldsymbol{B}$ que cada tipo de recurso $\boldsymbol{ri}$ contribui para a missão do agente.

O diagrama de recursos de uma missão corresponde a um diagrama de atividades UML (*Unified Modeling Language*) [12] que descreve as relações de precedência entre os recursos. Qualquer recurso em desacordo com este diagrama não traz nenhum benefício para a missão.

O diagrama de nodos é baseado no diagrama de recursos da missão. Cada recurso é substituído pelo(s) nodo(s) onde eles aparecem no sistema.

O cumprimento de uma missão $\boldsymbol{M}$ está relacionado à escolha de um itinerário $\boldsymbol{I}$. Para definição do itinerário são considerados:

- o tempo de computação para benefício máximo de cada tarefa $\boldsymbol{x_i}$, denominado $\boldsymbol{C_i}$;
- a latência de comunicação entre os nodos $\boldsymbol{N_k}$ e $\boldsymbol{N_y}$, denominada $\boldsymbol{L_{ky}}$;
- as dependências entre os recursos que aparecem na forma do diagrama de recursos da missão.

Devem ser analisados os custos para que o agente chegue ao nodo (vindo do nodo anterior) e para adquirir aquele recurso ($\boldsymbol{C_i + Q_i}$ onde: $\boldsymbol{C_i}$ é o tempo de execução no nodo e $\boldsymbol{Q_i}$ é o tempo na fila do processador local esperando para executar).

Para todo $\boldsymbol{r_i \in R}$, o benefício $\boldsymbol{B_{i,t}}$ obtido por utilizar o recurso $\boldsymbol{r_i}$ durante o intervalo de tempo $\boldsymbol{t}$ é dado por:

1) $\boldsymbol{r_i}$ do tipo *variable*

$$\boldsymbol{B_{i,t} = B_i * min(1, t/Ci)} \qquad \textbf{[1]}$$

onde $\boldsymbol{B_i}$ é o benefício máximo obtido de $\boldsymbol{r_i}$ e $\boldsymbol{C_i}$ é o tempo máximo de computação associado com $\boldsymbol{r_i}$ ($\boldsymbol{C_i}$ é o tempo que o agente deve ficar com o recurso para ganhar $\boldsymbol{B_i}$ e $\boldsymbol{t}$ é o tempo que ele realmente ficou).

2) r_i do tipo normal

$$B_{i,t} = B_i \quad \text{se} \quad t \geq C_i \qquad [2]$$
$$B_{i,t} = 0 \quad \text{se} \quad t < C_i$$

Nas equações acima são consideradas relações de precedência e de opcionalidade:

- se r_i precede r_j, e r_i não foi utilizado, então $B_j = 0$;
- se r_i' e r_i'' são réplicas de r_i, e r_i' já foi utilizado, então o benefício máximo de r_i'' passa a ser zero.

O benefício efetivo da missão é obtido através do somatório dos benefícios realizados a partir de cada recurso no itinerário:

$$B = \sum B_i(y) \quad \text{para todo} \quad r_i \in R. \qquad [3]$$

sendo $B_i(y)$ o benefício realizado pela utilização do recurso r_i que o agente visitou para o cumprimento de sua missão ao seguir o itinerário y, B é o benefício gerado pela utilização de todos os recursos da missão. O objetivo do algoritmo de definição de itinerário é maximizar o valor de B, respeitando o deadline D da missão.

O objetivo do algoritmo de definição de itinerário é maximizar o valor de B, respeitando o deadline D da missão.

Neste artigo é suposto que o diagrama de recursos vai sendo conhecido pelo agente móvel ao longo do caminho, em função dos dados que vão sendo obtidos nos nodos. Dessa forma, o agente móvel é definido como míope, ou seja, a cada momento ele conhece apenas os possíveis próximos nodos. Por conseguinte, o problema descrito neste capítulo não pode ser resolvido utilizando técnicas clássicas de otimização, pois na origem o diagrama de recursos não é conhecido; e durante o percurso os nodos visitados podem possuir capacidade de processamento limitada e podem não comportar os cálculos de otimização necessários. Exemplos desse tipo de ambientes podem ser encontrados em aplicações de supervisão de fábricas (agentes móveis na linha de produção [3, 13]).

Os agentes móveis carregam os resultados obtidos nos nodos visitados, mas com relação ao seu tamanho é assumido que o crescimento é pequeno, por este motivo o tamanho do agente não é considerado no modelo computacional.

IV. ALGORITMOS PARA DEFINIÇÃO DO ITINERÁRIO

Esta seção apresenta algumas heurísticas para a definição do itinerário do agente móvel. Elas são originalmente definidas em [4].

Neste modelo computacional para agentes móveis imprecisos com restrições temporais, cada agente possui certa flexibilidade na definição de seu itinerário. Esta flexibilidade está relacionada com características dos recursos. Cada heurística confere ao agente um comportamento distinto que, baseado nas diferentes características de cada recurso, é utilizado pelo agente móvel na definição do itinerário. Essas heurísticas podem ser utilizadas individualmente ou em pares/trios (através do uso de clones).

Em função da informação obtida através do último recurso recém-usado, o código de aplicação do agente identifica quais seriam os possíveis próximos recursos a serem visitados. Essa lista de destinos candidatos para o agente deve incluir as réplicas dos recursos candidatos, se essas existirem, e também a possibilidade de não executar nenhum deles, caso todos sejam opcionais. Caso a instância de recurso selecionada para ser a próxima a ser usada esteja em um nodo da rede diferente daquele onde o agente encontra-se, isto implicará no deslocamento do agente até aquele nodo. Seja qual for a decisão do agente, ela deve estar de acordo com o diagrama de nodos da missão, o qual define as possibilidades de recursos a serem utilizados.

- *Algoritmo Aleatório:* a escolha do próximo nodo a ser visitado pelo agente acontece randomicamente;
- *Algoritmo Preguiçoso:* procura executar os recursos o mais rapidamente possível, ignorando o benefício de cada recurso;
- *Algoritmo Guloso: s*empre que existir uma rota alternativa, será escolhido o recurso que apresentar o maior benefício para a missão;
- *Algoritmo Ponderado:* baseado na política AVDT (*Average Density Threshold*) [14] – elege como próximo recurso a ser executado aquele que apresentar a melhor relação benefício/tempo;
- *Versões com Relógio*: estimam o tempo disponível para executar os recursos restantes da missão. A folga existente é calculada pela diferença entre o deadline absoluto e o tempo atual, dividida pelo deadline relativo. Esta caracteristica acontece nos Algoritmos Guloso e Ponderado após cada recurso ser executado.

V. COMPARANDO HEURÍSTICAS

Com o objetivo de avaliar o desempenho das heurísticas simples, foram realizadas simulações com uma grande quantidade de diferentes cenários. Na simulação foi utilizado o modelo computacional descrito na seção III, considerando um sistema composto por 10 nodos e 15 recursos. Os recursos estão alocados de forma fixa nos nodos, sendo que alguns são replicados em outros nodos. A partir disso foram construídos os diagramas de nodos.

Foram definidos 10 segmentos de missão (diagramas de recursos) contendo, cada um, relações de precedência e alguns recursos [6]. Para cada segmento foi construído seu diagrama de nodos respectivo. A missão do agente é definida pela composição de 6 segmentos sorteados com reposição a partir deste conjunto de 10 diagramas. Este sorteio oferece 1.000.000 (10^6) possíveis diagramas de recursos diferentes, uma parte deste total foi utilizada, como será descrito posteriormente.

Para a simulação do tempo total da missão foram considerados: i) o tempo de fila do processador, com distribuição exponencial cujo valor médio varia de processador para processador com distribuição uniforme de 0 a 20; b) o tempo de uso do recurso, valor fixo por recurso, porém entre os recursos existe uma distribuição uniforme entre 1 e 15; c) o tempo de fila nos enlaces de comunicação, com uma distribuição exponencial, cujo valor médio varia de enlace para enlace entre dois nodos. O atraso médio dos enlaces varia entre 2 e 5 com distribuição uniforme; d) o tempo de comunicação necessário para transferir um agente móvel entre dois nodos quaisquer, fixo para todos os enlaces é igual a 1.

Espera-se que apenas alguns recursos sejam do tipo variável, desta forma iremos tratar esta questão através da discretização do recurso variável, criando 3 caminhos alternativos para fins de avaliação: execução completa, execução mediana e sem execução.

Após a execução de cada recurso, algum benefício é recebido e somado ao benefício total da missão. O objetivo da missão é alcançar o maior benefício respeitando o deadline especificado. Os benefícios adquiridos pela execução de cada recurso são fixos, sendo $B_0= 0$, $B_1=1$, $B_2=2$, $B_3=3$, $B_4=4$, $B_5=5$, $B_6=6$, $B_7=7$, $B_8=8$, $B_9=9$, $B_{10}=10$, $B_{11}=11$, $B_{12}=12$, $B_{13}=13$, $B_{14}=14$, $B_{15}=15$.

A qualidade do algoritmo (benefício global do algoritmo *G*) é calculada seguindo a seguinte equação:

$$G = \Sigma (B\_DA / NT) \qquad [4]$$

onde: *B_DA* é o benefício obtido pela execução dos recursos com deadline atendidos e *NT* é o número de tentativas.

Os recursos *r1*, *r2*, *r3*, *r4*, *r5*, *r6*, *r7*, *r8*, *r9* e *r10* são simples, isto é, são encontrados em um único nodo no sistema. Os recursos *r11*, *r12*, *r13*, *r14* e *r15* são replicados: existem instâncias destes recursos em dois ou mais nodos no sistema distribuído. A disposição dos recursos nos nodos é a seguinte: *r1* em {*N1*}; *r2* em {*N2*}; *r3* em {*N3*}; *r4* em {*N4*}; *r5* em {*N5*}; *r6* em {*N6*}; *r7* em {*N7*}; *r8* em {*N8*}; *r9* em {*N9*}; *r10* em {*N10*}; *r11* em {*N1,N2*}; *r12* em {*N3, N4*}; *r13* em {*N5,N6*}; *r14* em {*N7, N8*}; *r15* em {*N9, N10*}.

O simulador utilizado para a realização das simulações foi desenvolvido em Java, no grupo de pesquisa no qual o presente modelo computacional foi proposto.

A. Resultados da Simulação

O objetivo desta avaliação é observar e descrever distintamente o comportamento das heurísticas pelo intermédio de sucessivas simulações com diferentes configurações envolvendo diferentes situações. Buscou-se simulações abrangentes, foram lançados 100 agentes, com 100 diferentes configurações e 14 deadlines diferentes, o que resultou em 1400 execuções de cada configuração. Isto representa a execução de 140.000 missões usando cada uma das heurísticas.

Após a realização destas simulações, algumas afirmações podem ser feitas:

- O Algoritmo Preguiçoso sempre obteve os melhores índices com o menor deadline. Em todas as situações testadas, o Algoritmo Preguiçoso foi a primeira heurística a apresentar algum benefício (mesmo pequeno) quando o deadline era muito apertado;
- O Algoritmo Guloso sempre obteve o maior índice de benefício global em todas as situações testadas com deadline muito folgado;
- O Algoritmo Preguiçoso não apresentou bons resultados com deadlines folgados, seus índices de benefício global quando comparados as demais heurísticas foram pouco significativos;
- O Algoritmo Aleatório não apresentou um resultado satisfatório, o que já era esperado, pois seu fator de tomada de decisão é apenas um sorteio entre as próximas possíveis posições. Esta heurística, diferentemente das demais, não utiliza nenhum critério para sua tomada de decisão;
- O Algoritmo Ponderado e o Algoritmo Ponderado com Relógio apresentaram bons resultados com deadlines justos (maiores índices de benefício). Apresentaram bons resultados também com os demais tipos de deadline, geralmente resultados próximos aos melhores: em sua versão com relógio, alcançou valores próximos ao do Algoritmo Preguiçoso; em sua versão sem relógio, alcançou valores próximos ao Algoritmo Guloso e Guloso com Relógio;
- Cumprindo as expectativas, a versão com relógio do Algoritmo Guloso apresentou índices melhores, com deadlines apertados, quando comparada com a sua versão original.

Analisando o gráfico da Figura 1, percebe-se que a partir do momento em que os deadlines de todos os agentes da missão são atendidos, o benefício global obtido pelos algoritmos permanece estável, independente do tempo restante entre o deadline e o tempo gasto com a missão.

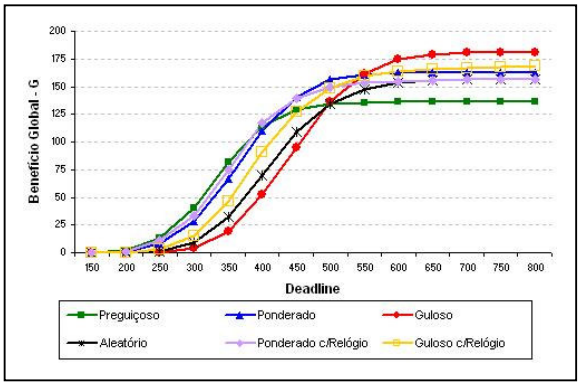

Fig.1. Benefício global das heurísticas simples

No gráfico acima, essas movimentações dos índices de benefícios adquiridos a cada deadline são facilmente perceptíveis. Por exemplo, com deadlines apertados, o Algoritmo Ponderado com Relógio apresentou melhores resultados que sua versão original, esta vantagem diminui até que o Algoritmo Ponderado (sem relógio) supera os valores de beneficio de sua versão com relógio. O mesmo pode ser observado com o comportamento dos Algoritmos Guloso e Guloso com Relógio.

A Figura 2(a) expõe os índices que indicam o percentual de agentes que obtiveram sucesso, referentes a cada deadline. Os algoritmos Aleatório e Guloso foram os que necessitaram de maiores deadlines para atingir 100% de atendidos.

Inicialmente, o Algoritmo Preguiçoso apresentou os maiores índices de atendimento, os algoritmos Ponderado e Ponderado com Relógio também obtiveram bons índices com deadlines justos, enquanto que o Algoritmo Guloso atingiu índices de atendimento mais baixos.

Versões com relógio podem conseguir 100% de deadlines atendidos com um determinado deadline e em outras tentativas não alcançarem este valor com deadlines maiores, esta oscilação é característica destas versões. As versões com relógio embutem um elemento de aleatoriedade em seus comportamentos o qual aparece nos resultados das simulações.

A Figura 2(b) exibe o gráfico com os tempos médios de resposta utilizados pelas heurísticas. Analisando o gráfico, nota-se que no Algoritmo Guloso há um maior consumo de tempo, o que comprova que esta heurística necessita de deadlines folgados para obter benefícios relevantes. As demais heurísticas movem-se alternadamente dentro de um intervalo de tempo com valores mais baixos.

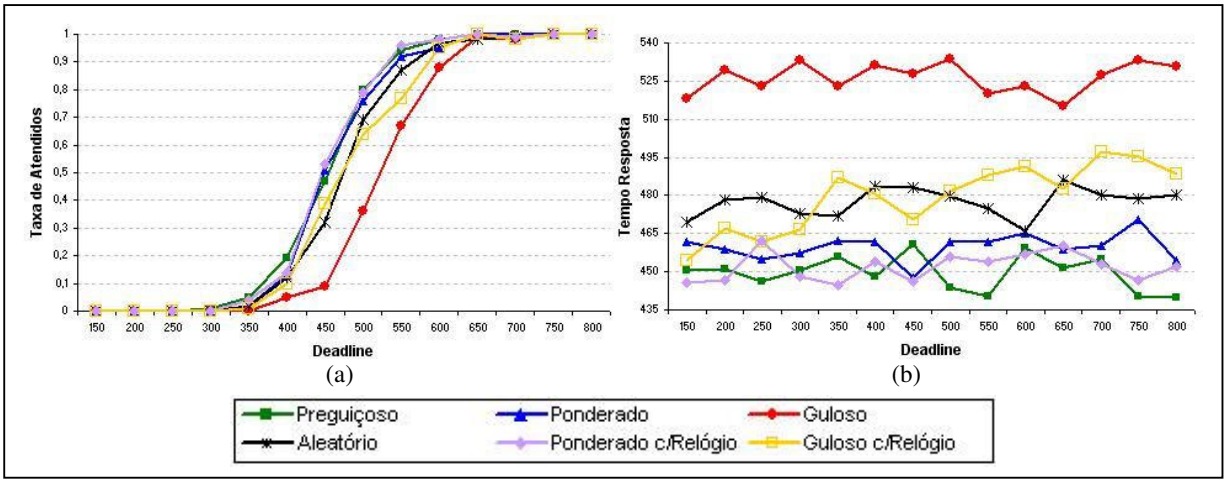

Fig.2. (a) Taxa média de deadlines atendidos das heurísticas simples e (b) tempo de resposta médio das heurísticas simples

VI. O MÉTODO DE PARES DE CLONES

Analisando os resultados das simulações descritas acima, foi observada a necessidade em se combinar as heurísticas para alcançar melhores índices de desempenho.

A miopia do agente pode limitar a eficiência das heurísticas existentes. Uma possível solução é o uso de um agente clone que adota uma segunda heurística.

Muitos fatores influenciam na decisão de qual é a melhor heurística simples. Estes fatores estão diretamente relacionados ao deadline da missão. Na tentativa em se obter os melhores resultados de uma forma geral, foi proposto o uso de agentes clones. No momento que a missão é iniciada, um par de agentes móveis utilizando diferentes heurísticas é lançado.

O par (composto pelo agente móvel original e o agente móvel clone) deixa o nodo origem seguindo diferentes itinerários em busca do mesmo objetivo. No final da missão ambos agentes voltam ao nodo inicial e então é feita uma comparação e são selecionados os melhores resultados individuais do par. Este método provavelmente aumentará o benefício global da missão e certamente não diminuirá seu valor global. No caso de um dos agentes não retornar ao nodo origem dentro do deadline, os dados obtidos pelo agente que já chegou serão utilizados. O overhead do sistema é controlado limitando o número de clones.

Como a capacidade computacional dos nodos do sistema pode ser pequena, a clonagem é feita uma única vez, no nodo origem. A partir deste momento os agentes viajam de forma independente e não existe uma relação de hierarquia entre eles.

O comportamento que cada agente deverá exibir ao longo de todo itinerário também é definido no nodo origem antes de sua partida. O uso de pares de agentes só foi possível porque as heurísticas simples apresentaram um comportamento distinto para cada faixa de deadline.

A. Resultados da Simulação

Nesta etapa de simulações foram lançados 100 agentes com 100 diferentes configurações testadas com 14 deadlines (que classificam-se como: apertados, justos e folgados). Os resultados discutidos a seguir são referentes a média destas execuções.

A dupla de algoritmos Aleatório/Aleatório (A-A) foi o único par lançado utilizando a mesma heurística devido ao seu comportamento irregular. O par de algoritmos Preguiçoso/Guloso (Pg-G) apresentou resultados satisfatórios, mas apenas compôs o envelope superior com deadlines folgados. Nos demais deadlines, seus resultados na maioria das vezes estiveram próximos aos índices mais altos de benefício adquirido.

O par de algoritmos Preguiçoso/Ponderado com Relógio (Pg-PdR), conquistou o melhor desempenho em situações nas quais o deadline estava apertado (até D=300). Pode–se afirmar que, para deadlines apertados, foi o par que apresentou melhores resultados, mas a medida que o deadline foi aumentado esta vantagem desapareceu.

É importante notar que o desempenho das duplas, em todas as situações, foi superior ao desempenho alcançado com a execução das heurísticas individualmente. As duplas que contiveram o Algoritmo Guloso garantiram altos índices em seus desempenhos com deadlines folgados.

O par de algoritmos Ponderado/Guloso (Pd-G) obteve destaque entre os demais para deadlines folgados e justos, mas apresentou resultados mais baixos para deadlines apertados. Na tentativa de suprir esta lacuna, foi lançado o par de algoritmos Ponderado com Relógio/Guloso (PdR-G). Se comparados os resultados entre estes dois pares, percebe-se a vantagem em se utilizar o segundo par, no qual a versão com relógio para o Algoritmo Ponderado trás um melhor desempenho para deadlines apertados e o bom desempenho apresentado pelo primeiro par com deadlines justos e folgados foi mantido.

O par Ponderado com Relógio/Guloso com Relógio (PdR-GR) atingiu resultados medianos, não apresentando destaque em nenhuma faixa de deadline.

A Figura 3 ilustra o comportamento das heurísticas (individualmente ou em pares), com seus respectivos benefícios globais obtidos. O envelope superior é formado pelos pares de algoritmos: Preguiçoso/Ponderado com Relógio, Preguiçoso / Ponderado e Ponderado/Guloso.

A Figura 3(a) apresenta todas as heurísticas simples e em clones que foram testadas. Para facilitar a visualização do comportamento das heurísticas, na Figura

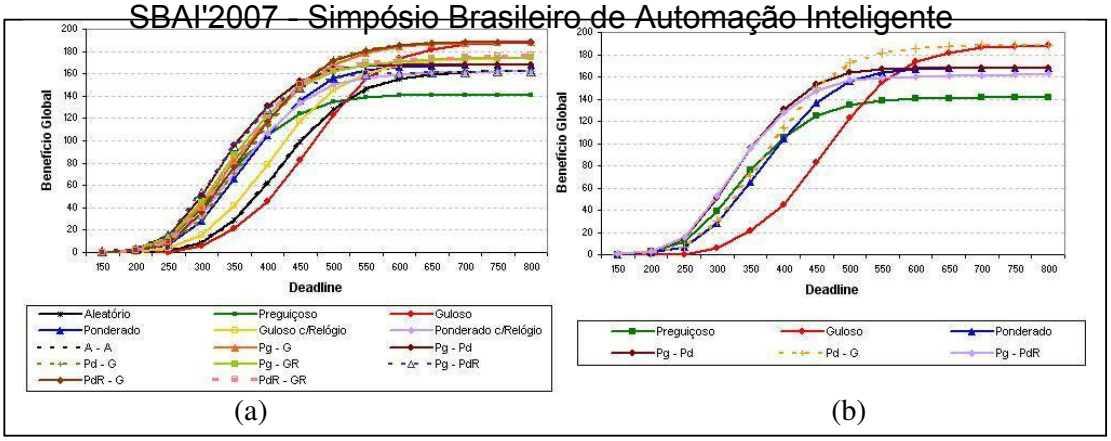

Fig. 3. Benefício Global das heurísticas simples e com clones

3(b) foram mantidas apenas as heurísticas que apresentaram os melhores desempenhos.

Em vista dos resultados encontrados, considerando o benefício global, pode-se concluir que o melhor par para deadlines apertados foi o par Preguiçoso/Ponderado com Relógio; para deadlines justos foi o par Preguiçoso / Ponderado; e para deadlines folgados foram os pares Ponderado/Guloso e Preguiçoso/Guloso.

Uma vez que o Algoritmo Preguiçoso supre os objetivos da missão para os menores deadlines, o Algoritmo Ponderado supre os objetivos da missão para os deadlines justos e o Algoritmo Guloso provê os melhores resultados para deadlines folgados, imaginou-se que a combinação entre eles obtivesse um comportamento que resultasse no cumprimento dos requisitos da missão para diferentes deadlines.

Dois trios de agentes foram lançados e, conforme esperado, obtiveram os maiores índices de benefício global relativo médio, uma vez que para compor os trios, foram utilizadas as heurísticas que geralmente formavam o envelope superior dos gráficos de benefício global.

VII. CONCLUSÕES

Este artigo propõe o uso em pares de heurísticas capazes de guiar o agente móvel através do sistema distribuído, conforme as premissas definidas no modelo computacional. Todas as heurísticas simples oferecem um baixo custo computacional, algo necessário, pois alguns nodos a serem visitados pelo agente móvel podem possuir baixa capacidade de processamento. Além disso, todas as heurísticas simples respeitam a premissa da miopia dos agentes móveis.

Existe a constante preocupação com o possível *overhead* provocado pela execução dos agentes clones, para isso o número de agentes clones por missão foi limitado. Assim, diminuindo a possibilidade que processamento extra para executar uma missão tenha como conseqüência piorar o desempenho dos demais agentes móveis que utilizam a rede.

Apesar das heurísticas discutidas neste artigo serem simples, pôde-se observar que elas apresentaram características bem distintas com relação ao seu comportamento frente aos diferentes tipos de deadlines.

Isto indica que o algoritmo a ser escolhido para uma nova missão deve estar de acordo com o deadline da missão.

Já que as heurísticas simples apresentaram um comportamento definido para cada faixa de deadline, é possível sugerir o comportamento mais adequado para uma nova missão, uma vez que o deadline da missão atual é conhecido. Considerando a possibilidade de se escolher a heurística a ser utilizada a cada nova missão, antes da partida do agente a heurística a ser utilizada pode ser cuidadosamente selecionada levando em conta apenas o deadline da missão atual e a experiência adquirida nas missões anteriores.

REFERÊNCIAS

[1] G. P. Picco, "Understanding, Evaluating, Formalizing, and Exploiting Code Mobility" PhD Thesis, Politecno di Torino, Italia, 1998.

[2] J.W. Baek, G.T. Kim, and H.Y. Yeom, "Timed Mobile Agent Planning for Distributed Information Retrieval" Computation and Engineering School, Seoul National University, Proceedings of AGENTS´01, Montreal, Quebec, Canada, Junho, 2001.

[3] M. Shin, M. Jung, and K. Ryu, "A Novel Negotiation Protocol for Agent-based Control Architecture", 5th International Conference on Engineering & Automation, Las Vegas, EUA, 2001.

[4] L. Rech, R. de Oliveira, and C. Montez, "Dynamic Determination of the Itinerary of Mobile Agents with Timing Constraints", IAT 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, pp. 45-50, Compiègne, França, Setembro, 2005.

[5] L. Rech, R. de Oliveira, and C. Montez, "A Clone-Pair for the Dynamic Determination of the Itinerary of Imprecise Mobile Agents with Firm Deadlines", ETFA 2006 11th IEEE International Conference on Emerging Technologies and Factory Automation, Setembro, 2006.

[6] L. Rech. "*Deteminação Dinâmica do Itinerário de Agentes Móveis Imprecisos com Restrições Temporais*". Tese de Doutorado, Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Brasil, 2007.

[7] K. Oida, and M. Sekido, "ARS: An efficient agent-based routing system for QoS guarantees", Elsevier Science, Computer Communications 23, pp. 1437-1447, 2000.

[8] O. Shehory, K. Sycara, P. Chalasani, *et.al*. "Agent Cloning: An Approach to Agent Mobility and Resource Allocation", IEEE Communications Magazine, pp. 58-67, July, 1998.

[9] S. K. Ong; W. W. Sun *Aplication of Mobile Agents in a Web-based Real-Time Monitoring System.* Int. J. Adv. Manuf. Technol. pp.33-40. 2003.

[10] J.W. Baek; G.T. Kim; H.Y. Yehom. *Timed Mobile Agent Planning for Distributed Information Retrieval* Escola de Computação e Engenharia. Seoul National University. Proceedings of AGENTS´01. Montreal. Quebec. Canadá. Junho. 2001.

[11] W. Qu, H. Shen, and Y. Jin, "Theoretical Analysis on a Traffic-Based Routing Algorithm of Mobile Agents" IAT 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, pp. 520-526, Compiègne, França, Setembro, 2005.

[12] J. Arlow, and I. Neustadt. *UML 2 and the Unified Process*. Second Edition, Practical Object-Oriented, Analysis and Design. Object Technologies Series, Addison-Wesley – Series Editors, 2005.

[13] S. Krishnamrthy, and I. Zeid, "Distributed and Intelligent Information Access in Manufacturing Enterprises through Mobile Devices", Journal of Intelligent Manufacturing, Kluwer Academic Publishers, Vol 15, pp.175-186, 2004.

[14] A. Davis, *et al*. "Flexible Scheduling for Adaptable Real-Time Systems", Proceedings IEEE Real-Time Tech. and App. Symp, pp. 230-239, Maio, 1995.