6
A Agência

A agência SIMPLES foi desenvolvida para negociar em leilões simultâneos de bens relacionados. Ela engloba um conjunto de técnicas para resolução dos vários subproblemas da negociação. Cada um dos seus componentes possui uma responsabilidade específica com relação aos subproblemas.

Para cada subproblema, aplicamos um conjunto de heurísticas a fim de identificar a mais adequada. Muitas vezes, a melhor heurística foi resultado da combinação de duas ou mais. Cada uma das técnicas aplicadas foi exaustivamente testada no ambiente do TAC.

A agência SIMPLES, aqui descrita, não faz lances apenas nas preferências dos seus clientes, nem emprega rebuscadas técnicas de IA. Ela aplica um conjunto de técnicas simples, que combinadas, mostraram um bom desempenho em cenários competitivos no ambiente do TAC.

6.1
Organização Multi-Agente

A agência envolve seis tipos de agentes. Os agentes são autônomos, assíncronos e cooperam entre si a fim de determinar os leilões em que desejam negociar, bem como a quantidade de bens e os preços a serem oferecidos. Os agentes interagem de forma independente, possibilitando descentralização do controle da agência. Como os agentes possuem uma visão limitada do estado total do mercado, o objetivo do protocolo de comunicação é manter a desempenho globalmente coerente dos agentes, sem violar a autonomia dos mesmos.

Esta seção descreve a arquitetura e os objetivos dos agentes desenvolvidos para a agência SIMPLES. Em seguida, detalhamos as tecnologias utilizadas para resolução do problema de leilões simultâneos de bens relacionados.

6.1.1
Sensor

Os *Sensores* são responsáveis por coletar informações do "mundo externo[18]" e por construir uma representação interna dos dados importantes, chamada Base de Conhecimento do Mercado. Os dados coletados por estes agentes são utilizados nas estratégias desenvolvidas em toda agência. Os sensores são seletivos no sentido em que coletam apenas as informações de relevância para uso dos demais agentes.

Os agentes Sensores são puramente reativos, pois respondem imediatamente às mudanças que ocorrem no ambiente quando alguma informação sobre os leilões é modificada. Por exemplo, quando o preço de cotação de algum bem aumenta ou diminui, os sensores são responsáveis por passar esta informação para a base de conhecimento.

Para o TAC, foram criados três agentes sensores: um para os hotéis, um para os vôos e um para os ingressos de diversão. Cada um supervisiona as mudanças que ocorrem nos leilões correspondentes e repassam as mudanças para a Base de Conhecimento do Mercado. Desta forma, este repositório mantém sempre as informações atuais do que ocorre no mundo externo.

6.1.1.1
Base de Conhecimento do Mercado

A Base de Conhecimento do Mercado é uma representação interna de todos os dados de interesse dos leilões. Nela, são inseridas as cotações dos preços de cada leilão, os bens já adquiridos pela agência SIMPLES, se o leilão já foi encerrado, entre outras informações que possam ser utilizadas na implementação das estratégias dos agentes.

Ela foi implementada com simples tabelas, porém sua implementação também é possível com outras estruturas de dados mais rebuscadas. As

informações armazenadas na Base de Conhecimento do Mercado estão disponíveis para toda a agência, sem a necessidade que todos os agentes se comuniquem com o meio externo. A interação dos agentes diretamente com os leilões virtuais pode causar uma demora indesejada, além de possíveis inconsistências dos dados. Por exemplo, o agente Negociador e Alocador precisam saber quantos bens já foram adquiridos em cada leilão. Esta informação deve ser consistente e atualizada.

Então, a Base de Conhecimento do Mercado fornece uma estrutura de dados atualizada pelos agentes Sensores, de fácil acesso para a utilização de toda agência SIMPLES.

6.1.2 Segmentador de Demanda

Os clientes são colocados nas classes “Fácil”, “Média” e “Difícil” de acordo com o período em que desejam viajar. O agente *Segmentador* de Demanda é responsável por fazer tal separação.

O Segmentador de Demanda é um agente reativo. Assim que o Sensor disponibiliza as preferências do cliente na Base de Conhecimento do Mercado, o Segmentador de Demanda posiciona cada cliente na classe correspondente. O agente Ordenador leva em consideração as preferências dos clientes “fáceis” na emissão das ordens de compra, bem como a dificuldade de aquisição dos bens dos clientes difíceis.

A segmentação de demanda desenvolvida para a agência foi específica para o domínio do TAC. Todavia, isto não invalida a existência de tal agente, uma vez que outras segmentações podem ser feitas em outros domínios.

[18] O mundo externo é constituído pelos leilões e clientes virtuais, como também por qualquer dado relevante ao mecanismo de negociação da agência.

6.1.3 Alocador

O *Alocador* faz a comunicação com o *solver* que soluciona o modelo de programação inteira. Ele passa os dados armazenados na KB do Mercado necessários para execução do modelo, que retorna um conjunto de bens que devem ser adquiridos pela agência. O conjunto de bens é formado por pacotes de viagens para os oito clientes da agência. Para facilitar o entendimento, chamaremos um conjunto de bens de um resultado do *solver*. Os resultados do *solver* são utilizados como base nas tomadas de decisões de compra e venda.

O *solver* recebe do Alocador, as preferências dos clientes, os bens já adquiridos pela agência até o momento e as cotações dos preços dos bens em cada leilão. O *solver* resolve o modelo, com os argumentos recebidos, e retorna o número de bens que deve ser negociado em cada leilão, bem como a pontuação associada ao conjunto de bens caso ele seja completamente adquirido. Se a agência consegue obter todos os bens sugeridos pelo *solver*, com os preços que foram passados, ela atingirá a sua pontuação máxima possível calculada pelo *solver*. O modelo inclui as restrições de compra dos bens relacionados. Dessa forma, o resultado do *solver* que fornece a pontuação máxima é composto apenas de pacotes viáveis de viagem.

Durante cada partida, o *solver* é acionado várias vezes não só em busca do resultado com a maior pontuação, mas das tantos maiores resultados. O modelo é executado, calculando vários conjuntos de bens que fornecem as maiores pontuações para uma mesma entrada de dados, ou seja, são calculados os melhores resultados para os mesmos valores num instante do jogo. A partir deste conjunto de resultados do *solver* foram utilizadas heurísticas para obtenção dos bens que são indispensáveis nas altas utilidades.

O Alocador é um agente reativo com estado interno. Ele percebe as mudanças ocorridas na Base de Conhecimento do Mercado e decide quando deve acionar o *solver* baseado nos dados anteriores. Ou seja, se ocorrerem mudanças significativas nos leilões que façam valer a pena a utilização do *solver*. Dessa forma, quando algum agente Sensor atualiza a KB do Mercado, ele notifica o Alocador que a avalia a necessidade de utilização do *solver*.

6.1.4
Supervisor

O *Supervisor* é responsável por iniciar a Agência. Ele espera pelo início dos leilões e envia uma mensagem para os demais agentes. Os agentes interagem de forma independente, possibilitando descentralização do controle da agência.

6.1.5
Ordenador

O agente *Ordenador* é responsável por enviar as diretivas de compra para os agentes Negociadores. Ele calcula as quantidades, preços e importância dos bens baseado nos resultados do *solver*.

Foram implementadas e testadas várias heurísticas para determinação de ordens de negociação pelo Ordenador. Os resultados do *solver* com as pontuações mais altas são uma rica fonte de informação para tomadas de decisão em negociação.

O Ordenador utiliza a freqüência com que os bens aparecem nos resultados do *solver* para emitir as ordens de compra e venda em cada um dos leilões. O capítulo seguinte possui os experimentos realizados com cada uma das heurísticas e o resultado obtido.

Além das quantidades, o Ordenador fornece os valores mínimo e máximo dos bens que os agentes Negociadores deverão fazer lances. Os valores mínimos são dados pelo preço corrente dos bens nos leilões. Apesar da possibilidade das ofertas de passagens e ingressos de diversão baixarem de preço, a agência faz lances a partir das cotações correntes. Quando o *solver* inclui um bem no seu resultado, ele está acreditando que o bem será adquirido pelo preço atual, trazendo um lucro para a agência, se estes forem arrematados por valores mais baixos.

Para calcular o valor máximo que pode ser oferecido pelos Negociadores para os leilões de hotéis, utilizamos mais uma vez o *solver*. Depois que o Ordenador obteve a quantidade de quartos que deseja numa noite em um determinado hotel, executamos o *solver* informando que os quartos foram

arrematados. O *solver* retorna a alocação com a utilidade associada. Em seguida, executamos novamente o *solver* informando que o mesmo leilão já encerrou e que os quartos não foram arrematados. Ele faz uma nova alocação com os leilões que ainda estão em aberto e calcula a nova utilidade. O valor do lance máximo dos hotéis é dado pela diferença entre a utilidade com e sem o mesmo bem. Ou seja, é o lucro marginal que o bem pode fornecer à agência.

Já para as passagens, o Ordenador emite ordens de compra com um pequeno incremento no valor mínimo. A agência obtém resposta imediata aos lances de passagens aéreas. Caso a agência não consiga arremata-las pelo valor corrente com uma pequena tolerância, o *solver* será acionado novamente para validar se ainda é lucrativa a aquisição da passagem pelo novo preço de oferta.

As ordens de compra e venda dos ingressos são emitidas apenas no final do jogo. Os lances de compra são feitos pelo valor médio do lucro que um ingresso proporciona aos clientes. Não é interesse da agência beneficiar os competidores comprando ou vendendo tickets aos demais se isto não for extremamente lucrativo para a mesma. Quando a agência compra um ingresso, ela fornece um lucro a um competidor. Suponha que um ingresso está sendo oferecido por $60 e o bônus que ele proporciona a um determinado cliente é de $100. Logo, o agente comprador terá um benefício de $40, enquanto que o vendedor terá de $60.

A agência só vende seus ingressos se eles não forem realmente utilizados pelos seus clientes pelo valor de $80 que segundo (Living, 2003), acredita-se ser o valor médio de fechamento dos leilões de tickets de diversão.

O Ordenador também atribui uma importância aos bens. Tal importância é influenciada pela análise da oferta e demanda. Ela implica que alguns produtos terão maior prioridade que outros na negociação em leilões.

Nesta seção fizemos uma descrição geral do agente Ordenador. No próximo capítulo, iremos apresentar as heurísticas utilizadas para emissão das quantidades, preços e importâncias dos bens.

6.1.6
Negociador

Os *Negociadores*, por sua vez, executam as diretivas do Ordenador. São esses agentes que se comunicam com os leilões, enviando lances ou oferecendo produtos. O Ordenador envia todos os parâmetros necessários para os Negociadores participarem dos leilões, os quais efetuam as diretivas de forma que maximize a sua performance.

Cada agente negociador recebe a categoria, o tipo e o dia de leilão que fará a negociação. O agente recebe também a quantidade de bens que deseja adquirir e um vetor com os preços mínimos e máximos que o agente está disposto a pagar pelos bens, correspondendo as quantidades. Por exemplo, o negociador está disposto a pagar por uma unidade um valor mínimo de $100 e máximo de $120, por duas unidades um mínimo de $90 e máximo de $115, por cada uma. Também é dada uma importância a cada bem.

Definimos duas categorias básicas de agentes Negociadores: reativos e adaptativos. Os Negociadores reativos recebem um valor de incremento além dos dados descritos acima. O Negociador começa a partida fazendo lances nos valores mínimos. Caso os lances não sejam contemplados, o incremento é adicionado ao valor do lance anterior. Este processo é repetido até que o lance seja aceito ou que o valor máximo seja atingido.

Já os negociadores adaptativos fazem um lance inicial com o valor mínimo. Caso o lance não seja aceito, um novo lance é submetido, calculado da seguinte maneira.

NovoLance = PreçoOferta + (ValorMáximo – PreçoOferta)*F

F = (PreçoOferta – LanceAnterior) / (ValorMáximo – LanceAnterior)

O preço de oferta é o preço corrente de venda do bem num leilão.

Novos lances são enviados até que um seja aceito, ou o valor máximo seja atingido. Os lances são enviados segundo um intervalo de atualização em função da importância do bem.

6.2 Segmentação

A separação dos clientes quantos às suas preferências de datas de viagem foi uma forma de beneficiar os clientes com necessidades mais simples de serem atendidas. Outras segmentações poderiam ser feitas baseadas em outras características, por exemplo, os bônus que os clientes oferecem por hotel. A agência poderia beneficiar os clientes que estão dispostos a pagar mais pelos seus pacotes de viagem.

A segmentação de pacotes não foi implementada até o momento de entrega desta dissertação. O objetivo deste tipo de segmentador é separar os conjuntos de pacotes de viagem pelas suas pontuações. Para isto, precisamos de um maior número de máquinas ou de processadores mais velozes, pois o tempo de negociação é curto para o caso do TAC. Portanto, nos preocupamos apenas em calcular o conjunto de soluções viáveis com altas pontuações, não sendo necessária a separação das mesmas.

A segmentação implementada pela agência SIMPLES foi específica para o TAC. Análises para outros domínios, diferente do TAC, devem realizadas, identificando possíveis classificações.

6.3 Modelo de Programação Inteira

A agência SIMPLES utiliza um modelo de programação inteira para auxiliar nos processos de tomadas de decisão. O modelo recebe as preferências dos clientes, os preços correntes e os bens já arrematados durante os leilões. Nós usamos o pacote comercial XPRESS como ferramenta de resolução do modelo. O *solver* calcula a alocação mais rentável dos bens para os clientes e retorna o número de bens que deve ser negociado em cada leilão e a pontuação associada a tal alocação.

O agente Alocador executa o modelo diversas vezes durante uma instância de jogo. Sempre que as cotações são atualizadas, o modelo é executado em busca do conjunto mais rentável de bens atualizado e sua pontuação correspondente.

O modelo de programação inteira utilizado pela agência foi definido por um conjunto de variáveis, restrições e uma função objetivo. A função objetivo maximiza a pontuação, definida pela utilidade menos o custo.

O modelo também foi usado para encontrar o conjunto de melhores resultados. Isto foi feito adicionando restrições cada vez que a melhor alocação foi calculada. A restrição adicionada é justamente o conjunto de bens da melhor alocação, ou seja, dado que aquela combinação de bens não é mais possível, o *solver* calcula a segunda melhor alocação. Da mesma maneira, para obter a terceira melhor, a melhor e a segunda melhor alocações novas restrições são adicionadas ao modelo.

A Tabela 6-1 mostra um exemplo com as três melhores alocações calculadas pelo *solver* e a pontuação correspondente. Para facilitar o entendimento, só apresentamos as alocações incluindo as passagens e as hospedagens. Para cálculo destas alocações, o modelo levou em consideração as preferências dos clientes e os preços atuais. Observe que alguns bens se repetem nas três alocações, enquanto outros variam.

Bens	1ª alocação	2ª alocação	3ª alocação
Ida 1	3	4	5
Ida 2	3	2	1
Ida 3	-	-	-
Ida 4	2	2	2
Volta 2	2	3	4
Volta 3	3	3	1
Volta 4	1	-	1
Volta 5	2	2	2
Hotel de Luxo 1	2	3	3
Hotel de Luxo 2	2	3	2
Hotel de Luxo 3	1	-	1
Hotel de Luxo 4	2	2	2
Hotel Barato 1	1	1	2
Hotel Barato 2	2	-	-
Hotel Barato 3	-	-	-
Hotel Barato 4	-	-	-
Pontuação	**3450**	**3412**	**3391**

Tabela 6-1: Exemplo de alocações.

O *solver* calcula a alocação ótima e normalmente faz isto bem rápido. Ele leva menos de um segundo para encontrar o melhor conjunto de pacotes de

viagem e sua pontuação. Todavia, desde que a programação linear inteira é NP-Completo, em alguns casos a resolução pode levar mais do que o tempo esperado. Portanto, um parâmetro de *timeout* foi configurado. Se o *solver* não pode encontrar a melhor alocação, ele retorna a melhor encontrada até a restrição de tempo dada.

6.3.1
Formulação de Programação Inteira

Fazemos aqui uma descrição detalhada da formulação de programação inteira para o problema da alocação, depois de apresentar o funcionamento e a finalidade do modelo.

6.3.1.1
Definição do Problema

Dadas as seguintes informações:

1. Disponibilidade em cada um dos dois hotéis, por noite;
2. Disponibilidade de ingressos para cada uma dos eventos, por noite;
3. Disponibilidade de passagens de ida, por dia;
4. Disponibilidade de passagens de volta, por dia;
5. Cotações dos preços para cada um dos dois hotéis, por noite;
6. Cotações dos preços das passagens de dia, por noite;
7. Cotações dos preços das passagens de volta, por noite.

Suponhamos dados também, um conjunto de clientes e que para cada cliente são conhecidas as seguintes informações:

1. Datas pretendidas para as viagens de ida e de volta;
2. Valor de um bônus a receber por hospedar o cliente no hotel de maior categoria;

3. Valor de um bônus a receber por fornecer ao cliente um ingresso para um evento (o valor do bônus depende de cada evento);

4. Valor de um bônus a receber por oferecer um pacote ao cliente.

Deve-se achar uma proposta de pacotes para os clientes que maximize a diferença entre os bônus ganhos e a penalidade paga por não cumprir as datas pretendidas para as viagens[19] e os custos dos pacotes. Observemos que a penalidade não é paga se não se oferece nenhum pacote ao cliente.

6.3.1.2 Formulação

O modelo aqui apresentado foi desenvolvido por (Liporace, 2001). A seguir, apresentamos a sua descrição detalhada.

A Tabela 6-2 mostra todos os dados passados para o modelo e o nome das variáveis associadas.

Número de clientes	NClientes	
Número de dias	NDias	
Hotéis disponíveis	H = {TT, SS}	
Eventos disponíveis	E = {AP, AW, MU}	
Disponibilidade hotéis	DPernoite(h,t)	$h \in H$; t = 1, … , NDias - 1
Disponibilidade passagens de ida	DIda(t)	t = 1, … , NDias - 1
Disponibilidade passagens de volta	DVolta(t)	t = 2, … , NDias
Disponibilidade ingressos	DTicket(t,e)	t = 1, … , NDias – 1; $e \in E$
Preços dos hotéis	PPernoite(h,t)	$h \in H$; t = 1, … , NDias - 1
Preços das passagens de ida	PIda(t)	t = 1, … , NDias - 1
Preços das passagens de volta	PVolta(t)	t = 2, … , NDias
Bônus Hotel de Categoria	HotelBonus(c)	c = 1, …, NClientes
Bônus Ingresssos	FunBonus(c,e)	c = 1, …, NClientes; $e \in E$
Data pretendida de ida	IAD(c)	c = 1, …, NClientes
Data pretendida de volta	IDD(c)	c = 1, …, NClientes

Tabela 6-2: Dados do problema

Observe que o hotel TT é o de maior categoria.

[19] Estas penalidades dependem do número de dias em que diferem as datas pretendidas e as datas propostas nos pacotes.

A Tabela 6-3 apresenta todas as variáveis binárias do modelo. Por exemplo, PERNOITE(TT, 1, 2) indica que o cliente 1 se hospeda no *Tampa Towers* Hotel na noite e.

PERNOITE(h, c, t)	h ∈ H; c = 1, … , NClientes; t = 1, … , NDias - 1
ESTADA(h, c)	h ∈ H; c = 1, … , NClientes
IDA(c, t)	c = 1, … , NClientes; t = 1, … , NDias – 1
VOLTA(c, t)	c = 1, … , NClientes; t = 2, … , NDias
TICKET(c, t, e)	c = 1, … , NClientes; t = 1, … , NDias – 1; e ∈ E

Tabela 6-3: Variáveis binárias.

Antes de escrever as descrições, analisamos uma quantidade que é útil na descrição do problema.

O somatório

$$\sum_{t=1}^{T} IDA(c,t) \quad (1)$$

para um dado cliente c e dia T fixos, vale 1 se o cliente c realiza a viagem de ida até o dia T e 0, caso contrário. De maneira similar, o somatório

$$\sum_{t=2}^{T} VOLTA(c,t)$$

vale 1 se o cliente c realiza a viagem de volta até o dia T e 0, no caso contrário. Portanto, a diferença

$$\sum_{t=1}^{T} IDA(c,t) - \sum_{t=2}^{T} VOLTA(c,t)$$

vale 1 se o cliente c pernoita no destino no dia T e 0 se não o faz.

As expressões a seguir definem as restrições do nosso modelo.

Cada cliente realiza no máximo uma viagem de ida[20].

[20] A viagem de ida não pode ser no último dia.

$$\sum_{t=1}^{NDias-1} IDA(c,t) \leq 1 \quad (2)$$

Existe uma restrição deste tipo para cada cliente c = 1, … , NClientes.

A viagem de volta só pode ser realizada após a viagem de ida.

$$VOLTA(c,T) \leq \sum_{t<T} IDA(c,t) \quad (3)$$

Existe uma restrição deste tipo para cada par c = 1, … , NClientes; T = 2, … , NDias.

O cliente volta se e somente se vai.

$$\sum_{t=1}^{NDias-1} IDA(c,t) = \sum_{t=2}^{NDias} VOLTA(c,t) \quad (4)$$

Existe uma restrição deste tipo para cada cliente c = 1, … , NClientes.

O cliente deve ficar em algum hotel todas as noites que passa no destino.

$$\sum_{h \in H} PERNOITE(h,c,T) = \sum_{t=1}^{T} IDA(c,t) - \sum_{t=2}^{T} VOLTA(c,t) \quad (5)$$

Existe uma restrição deste tipo para cada par c = 1, … , NClientes; T = 1, … , NDias - 1.

O cliente pode pernoitar apenas no hotel no qual está hospedado.

$$PERNOITE(h,c,t) \leq ESTADA(h,c) \quad (6)$$

Existe uma restrição deste tipo para cada tripla h $\in$ H; c = 1, … , NClientes; t = 1, … , NDias - 1.

O cliente é hospedado em apenas um hotel (i.e. não pode mudar de hotel).

$$\sum_{h \in H} ESTADA(h,c) \leq 1 \quad (7)$$

Existe uma restrição deste tipo para cada cliente c = 1, … , NClientes.

Cada cliente assiste cada evento uma vez, no máximo.

$$\sum_{t=1}^{NDias-1} TICKET(c,t,e) \leq 1 \quad (8)$$

Existe uma restrição deste tipo para cada par c = 1, … , NClientes; e ∈ E.

O cliente pode assistir eventos apenas quando estiver no destino e apenas um por noite.

$$\sum_{e \in E} TICKET(c,T,e) \leq \sum_{t=1}^{T} IDA(c,t) - \sum_{t=1}^{T} VOLTA(c,t) \quad (9)$$

Existe uma restrição deste tipo para cada par c = 1, … , NClientes; t = 1, … , NDias - 1.

O número de clientes que viaja cada dia está limitado pelo número de passagens.

1. Disponibilidade de passagens de ida.

$$\sum_{c=1}^{NClientes} IDA(c,t) \leq DIda(t) \quad (10)$$

Existe uma restrição deste tipo para cada data t = 1, … , NDias – 1.

2. Disponibilidade de passagens de volta.

$$\sum_{c=1}^{NClientes} VOLTA(c,t) \leq DVolta(t) \quad (11)$$

Existe uma restrição deste tipo para cada data t = 2, … , NDias.

O número de clientes que pernoitam em um hotel está limitado pela disponibilidade de vagas.

$$\sum_{c=1}^{NClientes} PERNOITE(h,c,t) \leq DPernoite(h,t) \quad (12)$$

Existe uma restrição deste tipo para cada par h ∈ H; t = 1, … , NDias - 1.

O número de clientes que podem assistir a um evento numa data determinada está limitado pela disponibilidade de ingressos.

$$\sum_{c=1}^{NClientes} TICKET(c,t,e) \leq DTicket(t,e) \quad (13)$$

Existe uma restrição deste tipo para cada par e ∈ E; t = 1, … , NDias - 1.

Em adição às restrições apresentadas, a cada iteração do modelo, uma nova restrição é atribuída para que um conjunto das melhores alocações seja calculado. A restrição adicionada é justamente o conjunto de bens da melhor alocação, ou seja, dado que aquela combinação de bens não é mais possível, o *solver* calcula a segunda melhor alocação. Da mesma maneira, para obter a terceira melhor, a melhor e a segunda melhor alocações são novas restrições adicionadas ao modelo.

A função objetivo é a diferença entre a soma dos bônus ganhos por hospedar clientes no hotel TT, aqueles obtidos com ingressos aos eventos e por oferecer um pacote viável, menos o valor das penalidades a pagar por não satisfazer as datas de viagem pretendidas pelos clientes, menos as despesas dos bens. A seguir descrevemos cada um dos termos na função objetivo.

1. Bônus por hotel de categoria:

$$\sum_{c=1}^{NClientes} HotelBonus(c) * ESTADA(TT,c)$$

2. Bônus por ingressos:

$$\sum_{c=1}^{NClientes} \sum_{e \in E} (FunBonus(c,e) * \sum_{t=1}^{NDias-1} TICKET(c,t,e))$$

3. Bônus por oferecer pacote:

$$1000 * \sum_{c=1}^{NClientes} \sum_{h \in H} ESTADA(h,c)$$

4. Penalidade por datas diferentes:

$$100*[(\sum_{T<IAD(c)}\sum_{t=1}^{T} IDA(c,t) + \sum_{T>IAD(c)}(1-\sum_{t=1}^{T} IDA(c,t)) +$$
$$(\sum_{T<IDD(c)}\sum_{t=2}^{T} VOLTA(c,t) + \sum_{T>IDD(c)}(1-\sum_{t=2}^{T} VOLTA(c,t))) +$$
$$\sum_{c=1}^{NClientes}((IAD(c)+IDD(c)-2*NDias)*(1-\sum_{h\in H} ESTADA(h,c)))]$$

5. Custo da viagem:

$$\sum_{t=1}^{NDias-1} DIda(t)*PIda(t) +$$
$$\sum_{t=1}^{NDias-1} DVolta(t)*Volta(t) +$$
$$\sum_{h\in H}\sum_{t=1}^{NDias-1} DPernoite(h,t)*Pernoite(t)$$

A seguir discutimos brevemente cada parte da expressão de penalidade por datas diferentes:

Lembremos que o somatório da equação (1)

$$\sum_{t=1}^{T} IDA(c,t)$$

vale 1 se o cliente c realizou a viagem de ida até a data T. Ou seja, no começo vale 0 e muda o seu valor para 1 na data que ele viaja. Assim o somatório

$$\sum_{T<IAD(c)}\sum_{t=1}^{T} IDA(c,t)$$

conta, para um dado cliente c que realizou a viagem de ida antes da data desejada IAD(c), o número de dias que esta viagem foi adiantada. Observe a Figura 6-1.

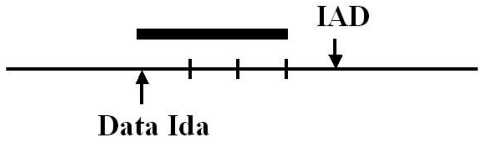

Figura 6-1: Viagem de ida antes da data desejada.

Analogamente, o somatório

$$\sum_{T>IAD(c)} (1-\sum_{t=1}^{T} IDA(c,t)) \quad (14)$$

conta, para um dado cliente c que realizou a viagem de ida após a data desejada IAD(c), o número de dias que esta viagem foi atrasada. Observe a Figura 6-2.

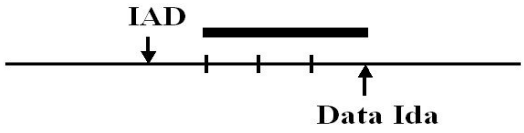

Figura 6-2: Viagem de ida após a data desejada.

De maneira similar os somatórios

$$\sum_{T<IDD(c)} \sum_{t=2}^{T} VOLTA(c,t)$$

e

$$\sum_{T>IDD(c)} (1-\sum_{t=2}^{T} VOLTA(c,t)) \quad (15)$$

contam os dias que a data real da viagem de volta se afasta da data pretendida.

Observemos que se o cliente não realiza nenhuma viagem, os somatórios (14) e (15) acrescentam uma penalidade, por cada cliente c que não viaja, de valor:

$$(NDias - IAD(c)) + (NDias - IDD(c)) = -(IAD(c) + IDD(c) - 2 * NDias)$$

O último termo

$$\sum_{c=1}^{NClientes} ((IAD(c) + IDD(c) - 2 * NDias) * (1 - \sum_{h \in H} ESTADA(h,c)))$$

tem como finalidade compensar esta penalidade que não deve ser paga caso o cliente não viaje. Observe.

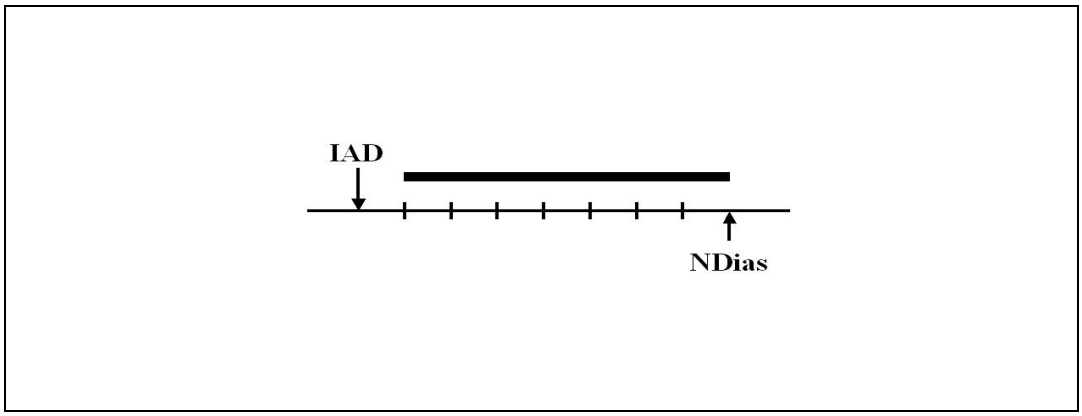

Figura 6-3: Penalidade artificial se o cliente não viaja.

6.4
Ordens de Compra

O Ordenador é o agente responsável pela emissão das ordens de compra. Ele passa as diretivas para o Negociador que interage diretamente com os leilões. Para as tomadas de decisão, ele utiliza os resultados do *solver* e a segmentação de demanda, além de levar em consideração os protocolos de negociação.

O Ordenador emite ordens de compra imediatas para os clientes que desejam viajar no dia 1 e voltar no dia 2 e para os que querem partir no dia 4 e voltar no dia 5, isto é, para os clientes “fáceis”.

Para determinar as quantidades dos bens dos demais clientes, o Ordenador utiliza o *solver* para auxilia-lo. Todas as heurísticas experimentadas aqui levaram em consideração a freqüência com que cada bem aparece nas alocações.

Uma primeira heurística implementada calcula as quantidades de cada um dos bens pela média em que aparece nas alocações. Utilizando a Tabela 6-1 como exemplo, o Ordenador emitiria a ordem de compra de quatro passagens de ida para o dia 1, já que este valor é a média das quantidades nas três alocações.

Os experimentos realizados, apresentados no próximo capítulo, mostram que a média gerava desperdícios na compra de hospedagens desnecessárias. Outra heurística implementada e testada utiliza a mediana. Neste caso, o Ordenador começa enviando ordens de compra para o primeiro decil. Em seguida, ele aumenta as quantidades para o primeiro quartil e enfim, faz ofertas na mediana. Isto faz com que os Negociadores tentem adquirir poucos bens no início da instância, evitando o prejuízo.

6.4.1 Bens Abundantes

A importância dos bens determina o intervalo de negociação dos mesmos. Isto é, o tempo em que um novo lance incrementado será enviado ao servidor de leilões pelos Negociadores. O Ordenador considera a análise da demanda realizada para determinação da importância dos bens.

Para os leilões de hotéis de luxo nos dia 1 e 4, quando normalmente existe quartos disponíveis para a demanda, é dada uma importância média. Isto significa que os lances serão atualizados cinco vezes antes de atingir o valor máximo.

Os leilões de hotéis baratos possuem uma importância baixa mesmo nos dias mais concorridos. O valor do lance pode ser incrementado até nove vezes para atingir o seu limite, isso se o leilão não for encerrado antes.

As passagens aéreas também representam bens abundantes. Porém, elas levam uma alta importância, uma vez que, é desejado saber rapidamente se é possível arremata-las.

As passagens para os clientes “fáceis” são negociadas no segundo minuto de jogo, logo que as primeiras cotações são disponibilizadas. As demais passagens são adquiridas na medida em que os leilões de hotéis fecham e o agente obtém resposta sobre a arrematação dos quartos. As últimas passagens são apenas adquiridas no último minuto de jogo quando todos os leilões de hotel já foram encerrados.

6.4.2 Bens Escassos Estratégicos

Os hotéis de luxo dos dias 2 e 3 representam bens estratégicos na formação de pacotes de viagem. Tais bens são considerados como os mais disputados em todo o conjunto de leilões. Por isso, foi dada uma alta importância aos mesmos.

Segundo a análise da demanda, os ingressos de diversão também são escassos e, por isso, recebem uma alta importância. Outro fator que interfere na sua importância é o fato de eles serem apenas negociados no final da instância de jogo, quando não há muito tempo para a negociação.

Todas as importâncias atribuídas aos bens foram demasiadamente testadas. O capítulo de experimentos apresenta a exaustiva trajetória na calibração de tais variáveis.

6.5 Estratégias de Negociação

Depois que as ordens de compra são definidas pelo Ordenador, cabe aos negociadores efetua-las. São instanciados 28 agentes negociadores, um para cada leilão, que implementam duas estratégias de negociação: uma reativa e uma adaptativa. Cada uma das heurísticas foi testada separadamente e a que apresentou melhor resultado foi incorporada à versão atual da agência.

6.5.1 Reativa

A primeira heurística testada foi a reativa. Neste caso, assim que o Negociador recebe os parâmetros necessários para negociar, ele começa a fazer lances nos valores mínimos. Quando os sensores obtêm a resposta sobre o lance, eles notificam o Negociador. Então, o Negociador interpreta a resposta que pode ter três status distintos, são eles:

1. O lance foi completamente aceito. Neste caso, o Negociador espera por novas ordens. Se o agente for negociador de hospedagens, ele fica acompanhando o leilão para observar se o lance permanece entre os dezesseis vencedores. Caso o status mude, ele toma a atitude correspondente aos status 2 ou 3.

2. O lance foi parcialmente aceito. Quando isto acontece, o Negociador submete uma nova oferta apenas para os bens que não foram adquiridos. O novo valor do lance é dado pelo valor do vetor de preços adicionado de um incremento. Vale lembrar que o Negociador possui um vetor com valores dos lances por quantidade de bens. Isto torna fácil a obtenção dos preços dos lances dos bens ainda não arrematados, uma vez que estes valores já foram calculados pelo Ordenador.

   O valor do incremento é um parâmetro adicional que este tipo de agente recebe do Ordenador.

3. O lance foi completamente rejeitado. Neste caso, o agente simplesmente emite um novo lance para a mesma quantidade de bens. O novo valor do lance é dado pelo valor anterior adicionado do incremento.

O Ordenador pode aumentar o número de bens que deseja adquirir em um determinado leilão. Quando isto ocorre o Negociador faz uma nova oferta ou simplesmente substitui o lance anterior, caso ele ainda não tenha sido processado.

O agente também pode incrementar o lance antes de obter uma resposta. O Negociador possui um parâmetro que é a importância do bem que determina um intervalo máximo de tempo para a submissão de um novo lance. Isto foi feito para evitar que bens sejam perdidos para os competidores mesmo antes da agência

atingir o valor do lance máximo permitido, especialmente no caso dos hotéis que fecham aleatoriamente.

6.5.2 Adaptativa

Os Negociadores adaptativos possuem basicamente a mesma heurística que os reativos. A principal diferença é a forma como os preços dos lances são incrementados. Ele não possui o argumento de incremento. Ele adapta o seu lance, calculando o novo valor a partir do valor anterior e do preço de cotação do leilão. A fórmula utilizada para este cálculo foi apresentada na seção 6.1.6. O novo valor é multiplicado por um fator F que é uma razão do preço de cotação pelo valor máximo permitido.

A adaptação em função da cotação permite que os lances sejam mais reais. Ou seja, os valores ficam mais próximos do mercado. Além disso, caso a cotação do bem seja maior que o valor máximo permitido, o Ordenador é notificado para decisão da compra de um outro conjunto de bens, já este não é mais viável.

O agente adaptativo também sofre interferência da importância no intervalo de submissão de um novo lance.

Durante os testes o fator F foi diversas vezes alterado em busca de um melhor desempenho da agência. A performance dos agentes é avaliada nos testes.

6.6 Aprendizado

O desenvolvimento da agência foi constantemente supervisionado. Alguns parâmetros, como a importância dos bens, intervalo de atualização dos lances, quantidade de bens a ser negociado no início do jogo, valores dos lances, entre outros, foram ajustados na medida em que testes foram realizados. Portanto, o aprendizado da agência ocorreu de forma supervisionada com auxílio humano que avalia os testes e calibra os parâmetros. Os ajustes feitos nos parâmetros são mostrados no capítulo dos experimentos, acompanhados dos resultados.

A Figura 6-4 mostra o processo realizado na configuração dos parâmetros da agência.

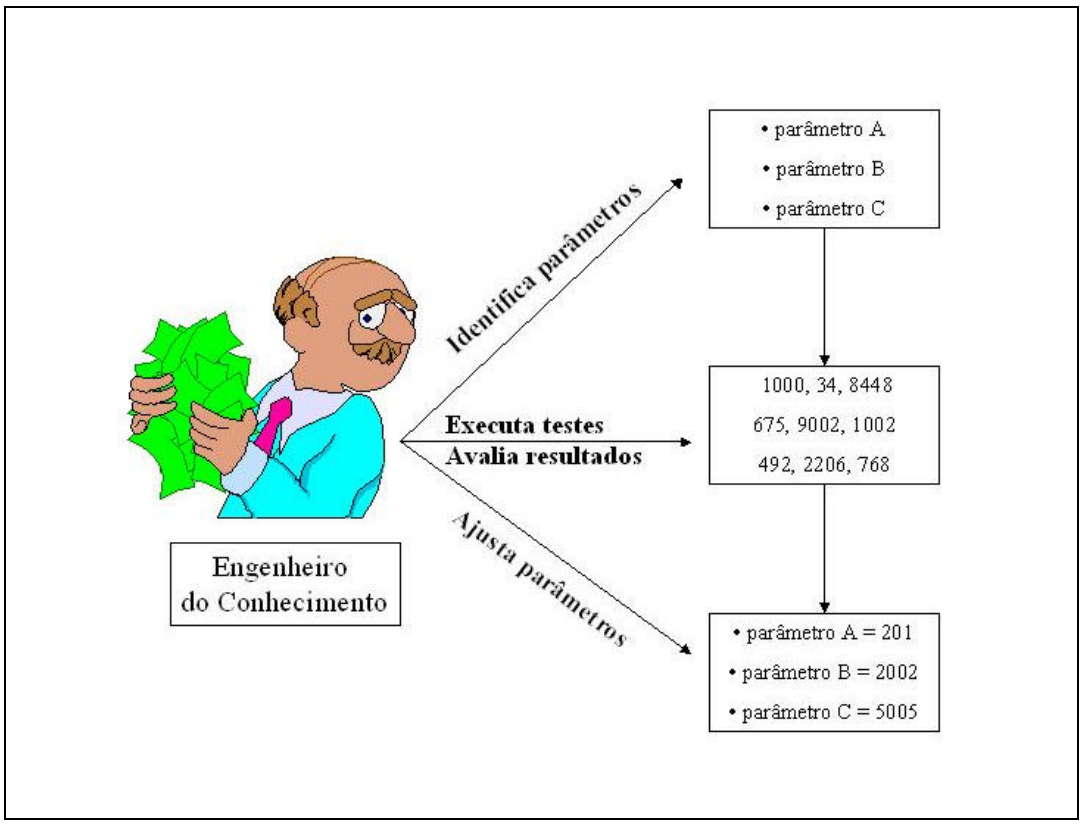

Figura 6-4: Processo de aprendizado.

6.6.1 Ambiente

O ambiente de aprendizado e treinamento dos agentes é composto por um repositório de agentes, um servidor de leilões e uma base de conhecimento, conforme mostrado na Figura 6-5. O usuário da ferramenta de treinamento define um plano de treinamento para o agente. Para isto, escolhe os adversários que compõem cada disputa e o número de instâncias de jogo.

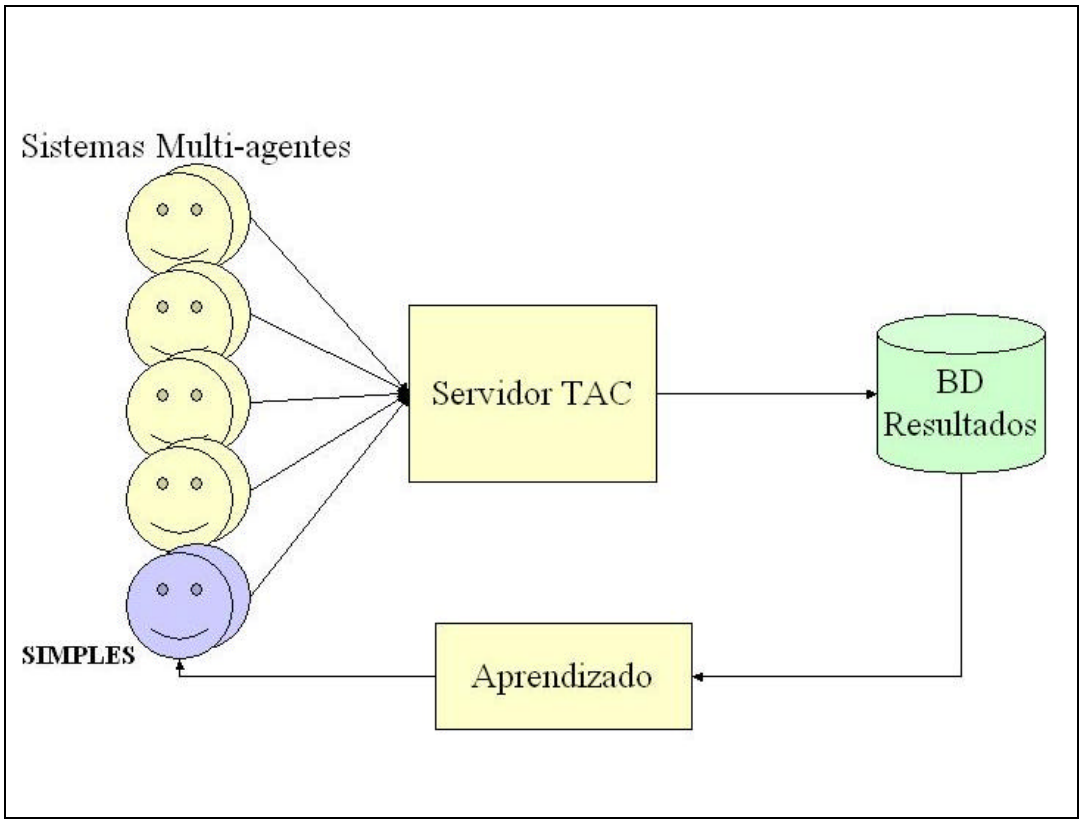

Figura 6-5: Ambiente de treinamento da agência SIMPLES.

No final de cada instância, os resultados são armazenados numa base de dados que ficam disponíveis para consultas. Antes do início de um novo treinamento, um procedimento de aprendizagem é realizado com base nos resultados das partidas anteriores. Este processo é repetido inúmeras vezes até que um resultado satisfatório seja obtido. A inclusão do um algoritmo de aprendizagem automática é uma possível extensão deste trabalho.

6.6.1.1 Agentes

Além da agência SIMPLES, dois agentes compõem o ambiente de aprendizado: *LivingAgentLike* e *DummyAgent*. O *LivingAgentLike* é similar ao *LivingAgent*, o campeão da segunda edição do TAC. Implementamos o *LivingAgentLike* a partir da descrição fornecida por (Klaus, 2001). Já o

DummyAgent[21] é um agente que acompanha o servidor TAC. Ele foi desenvolvido para preencher as instâncias de jogo quando oito participantes não se inscrevem. O *DummyAgent* possui uma estratégia simples, fazendo altos lances apenas nas preferências dos seus clientes. A descrição da estratégia do agente Dummy é fornecida na seção 7.1.

Combinações de *LivingAgentLike* e *DummyAgent* foram feitas com o intuito de avaliar o desempenho da agência, como também de melhor ajustar os seus parâmetros.

6.6.1.2 Informações Disponíveis

A Tabela 6-4 exibe os dados disponíveis para o sistema de aprendizado. No final de cada partida, as informações obtidas pelo servidor e pelo agente são inseridas numa base de dados que o engenheiro do conhecimento pode consultar para aprendizado da agência. As informações adquiridas pelo servidor não são visíveis ao agente durante a partida e só podem ser usadas quando a mesma é encerrada.

Período \ Visão	Servidor	Agente	Ambos
Início	Preferências dos agentes	Preferência do agente	ID e hora da partida
	Alocação dos agentes	Alocação do agente	
Durante	Transações efetuadas por cada agente (agente, hora, quantidade, preço).	Evolução dos preços nos leilões	Leilões em andamento ou fechados
Final	Pontuação de cada agente (utilidade, custo, score e alocação).		
	Transações efetivadas em toda partida (hora, leilão dia, agente, bem, quantidade e preço).		

Tabela 6-4: Dados disponíveis ao ambiente de aprendizado.

[21] Nome escolhido pelos organizadores do TAC.