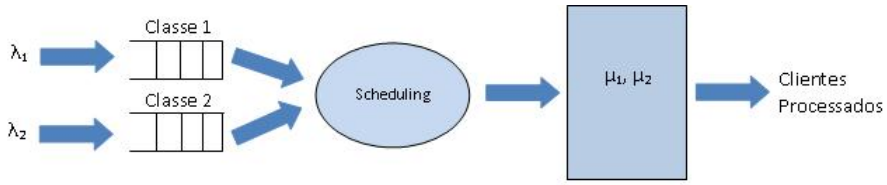
INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa



Sequenciamento em filas de servidor único.

Variante da regra μc.

Zita Daniela Batista Fernandes

Dissertação para obtenção do Grau de Mestre em

Engenharia Electrotécnica e de Computadores

Júri

Presidente: Doutor Francisco Miguel Prazeres Silva Garcia

Orientador: Doutor Carlos Filipe Gomes Bispo

Doutor João Manuel Freitas Xavier

Doutor Carlos Baptista Cardeira

Outubro de 2007

Abstract

The goal of the work described in this thesis concerns the presentation of arguments that bring a new perspective to some existing results of queuing networks. Comparison between numerical data obtained with models that use only buffer length as state variable with models that use also server state show that the models which only use buffer length do not correctly represent the systems under study. Also a new scheduling policy is presented. It is motivated by the belief that the marginal cost of waiting should not be a constant and by the belief that strict priorities are only interesting to customers having higher priority. With this new policy we propose to replace strict priorities, as is the case of the $\mu c-$rule. The variance of the cycle time for all classes achieved with the new policy will be presented, as is numerically computed. The tools used to get the results presented in this work, such as Dynamic Programming and Discrete Event Simulation, are also described.

Keywords

Queuing Networks, Scheduling Problem, Queuing Networks Modeling, State Costs, Variance of Waiting Time.

Resumo

O objectivo do trabalho descrito nesta dissertação prende-se com a apresentação de argumentos que dão uma nova perspectiva a alguns resultados clássicos da teoria das filas de espera. A comparação entre dados numéricos obtidos com modelos que fazem uso apenas do número de clientes no sistema como variáveis de estado e com modelos que também usam o estado dos servidores mostra que os modelos que apenas usam o número de clientes não representam os sistema em estudo de forma correcta.

Também se apresenta uma nova política de sequenciamento. Esta é motivada pela crença de que o custo marginal de esperar não deveria ser constante e pela crença de que as prioridades estritas, como forma de gerir filas de espera, são interessantes apenas para os clientes que pertençam à classe mais prioritária. Com esta nova política propõe-se substituir prioridades estritas, como é o caso da regra μc.

A variância do tempo de ciclo para todas as classes que se obtém com a nova política é apresentada, através de exemplos numéricos.

Os instrumentos utilizados para obtenção dos resultados apresentados ao longo da dissertação, como sejam a Programação Dinâmica e a Simulação de Sistemas Dinâmicos de Eventos Discretos, são também descritos.

Palavras-Chave

Redes de Filas de Espera, Políticas de Sequenciamento, Modelação de Redes de Filas de Espera, Custos de Estado, Variância do Tempo de Ciclo.

Conteúdo

1 Introdução **1**

1.1 Estrutura da dissertação 2

2 Enquadramento **5**

2.1 Redes de Filas de Espera 5

2.1.1 Modelos de Filas de Espera 5

2.1.2 Modelos para as Chegadas e Processamentos 6

2.1.3 Parâmetros Estruturais 7

2.1.4 Políticas de Operação 7

2.1.5 Estabilidade das Redes de Filas de Espera 9

2.1.6 Sequenciamento 10

2.2 Cadeias de Markov 12

2.2.1 Cadeias de Markov em Tempo Discreto e Estado Discreto 12

2.2.2 Cadeias de Markov em Tempo Contínuo e Estado Discreto 13

2.2.3 Uniformização 15

2.2.4 Problemas de Decisão de Markov 16

2.3 Programação Dinâmica 20

2.3.1 Formulação de Problemas 20

2.3.2 O Algoritmo de Programação Dinâmica 21

2.3.3 Problema de Horizonte Infinito 22

2.3.4 Iteração de Valor 23

3 Formulação de Problemas **25**

3.1 Modelação usando 1 Variável VS 2 Variáveis 25

3.1.1 Modelação usando 1 variável 26

3.1.2 Modelação usando 2 variáveis 29

3.1.3 Comparação dos Modelos 31

3.2 A $\mu\Delta$-Rule 34

4 Simulação **37**

4.1 Características Gerais 37

4.2 Modelação da Rede 38

4.3 Dinâmica da Rede 40

4.4 Especificação das Políticas 41

4.5 Arquitectura do Simulador 41

4.6 Recolha de Informação 43

5 Resultados **45**

5.1 Modelação usando 1 Variável VS 2 Variáveis 45

5.1.1 Programação dinâmica 46

5.1.2 Simulação da μc-rule 47

5.1.3 Comparação de Resultados 49

5.2 A $\mu\Delta - Rule$ 50

5.2.1 Resultados da PD 50

5.2.2 Formulação Intuitiva da Política 51

5.2.3 Variância do Tempo de ciclo 53

5.2.4 Variância Global e Dependência dos Processos 55

6 Conclusões e Desenvolvimentos Futuros **57**

6.1 Desenvolvimentos Futuros 59

A *Unified Modelling Language* do simulador. **61**

Lista de Figuras

2.1 Exemplo de uma rede de filas de espera com um servidor e uma classe de clientes. . . . 5
2.2 Rede de filas de espera com três classes de clientes. 8
2.3 Sequenciamento de três classes de clientes para um servidor. 9
2.4 Exemplo de controlo de sequenciamento aplicado a uma rede com duas classes de clientes. 11

3.1 Rede de filas de espera com um servidor e duas classes de clientes. 26
3.2 Diagramas de transição de estado em tempo contínuo. 27
3.3 Diagramas de transição de estado uniformizados. 28
3.4 Diagrama de Transições do modelo de duas variáveis em tempo contínuo. 29
3.5 Diagrama de Transições do modelo de duas variáveis uniformizado. 30
3.6 Matriz de decisão da μc-rule e transições possíveis. 32

4.1 Rede de filas de espera. 40
4.2 UML. 42
4.3 Interface. 42
4.4 Tabela de output do simulador. 43

5.1 Custo de estados iniciais do espaço de estados. 46
5.2 Custo de estados centrais do espaço de estados. 47
5.3 Ficheiro de entrada. 47
5.4 Rede de filas de espera virtual criada na simulação. 48
5.5 Intervalo de confiança a 95% para estados inciais do espaço de estados. 48
5.6 Intervalo de confiança a 95% para estados inciais do espaço de estados. 49
5.7 Sobreposição dos custos obtidos por PD no intervalo de confiança. 49
5.8 Sobreposição dos custos obtidos por PD no intervalo de confiança. 49
5.9 Matrizes de decisão. 51
5.10 Matriz de decisão genérica da $\mu\Delta - rule$. 53
5.11 Matrizes de decisão e resultado teórico. 54
5.12 Sobreposição dos custos obtidos por PD no intervalo de confiança. 54

5.13 Variância do tempo de ciclo global. 55
5.14 Dependência da variância do tempo de ciclo da classes de clientes. 56

A.1 UML. 62

Capítulo 1

Introdução

Os sistemas de redes de filas de espera constituem um importante instrumento de modelação em áreas tão diversas como sejam os sistemas de manufactura ou a transmissão de dados. Decorrentes destas aplicações, diversos resultados derivados pela teoria das redes de filas de espera têm sido aplicados a sistemas reais de modo a que o comportamento dos mesmos possa ser controlado adequadamente. O comportamento do sistema pode ser avaliado segundo várias perspectivas, como sejam os processos pelos quais são seleccionados os clientes a serem atendidos ou o modo como estes são encaminhados na rede. Dependendo do propósito, uma rede de filas de espera poderá ser estudada desde da sua estrutura até ao modo como os diferentes constituintes da mesma interagem.

No que diz respeito ao modo como os constituintes da rede interagem, vários tipos de políticas de operação podem ter-se em conta. No caso do presente trabalho, pretende-se estudar o sequenciamento em filas de servidor único.

As políticas de sequenciamento têm como objectivo gerir o modo pelo qual os clientes contidos nos buffers de uma rede são processados partindo de informação contida na mesma rede. Estas políticas podem ainda ser denominadas de prioridade estrita quando a política tem como propósito servir os clientes de acordo com uma lista de prioridade, servindo os clientes das classes menos prioritárias apenas na ausência de clientes das classes mais prioritárias. Do ponto de vista desta tese, considera-se que este tipo de políticas poderá não ser a forma mais adequada de escolher que clientes servir. Entende-se que se deveria ter mecanismos de prioridade com uma maior equidade no acesso a serviço por parte das classes menos prioritárias, contrariamente ao que acontece.

Partindo do pressuposto anterior, optou-se por elaborar uma variante do problema clássico de sequenciamento em filas de espera que deu origem à conhecida μc-rule. Nesta política a máxima prioridade é atribuída à classe de clientes i, que tiver um maior valor de $\mu_i c_i$, onde c_i corresponde à taxa

de custo linear atribuído à classe i e μ_i à taxa média de processamento de clientes da classe i. No caso da nova variante, a prioridade das classes deverá ser atribuída de acordo com um factor dado por $\mu_i\,(a_i + 2x_i b_i - b_i)$, onde a_i e b_i correspondem a taxas de custo linear e quadrático, respectivamente, μ_i corresponde à taxa de processamento de clientes da classe i e x_i ao número de clientes dessa classe no sistema. Esta nova política permite favorecer as classes não prioritárias através do aumento da sua prioridade relativamente à classe prioritária, como função do estado do sistema. Finalmente, reportando ainda à nova política determinada, foi posta a hipótese de se poder obter uma variância total do tempo de permanência no sistema mais baixa do que no caso de ser usada a $\mu c-$rule. A nova política pode ser vista como possuindo um mecanismo de suavização de prioridades, em oposição às tradicionais prioridades políticas de estritas. Ou seja, existem estados do sistema para os quais as classes não prioritárias obtêm acesso ao servidor mesmo na presença de clientes das classes mais prioritárias.

Decorrente da análise de diversa bibliografia relacionada com o tema anterior, foi ainda possível a formulação de um novo problema que nada tem a ver com a análise da performance de políticas mas sim com o modo como os sistemas são modelados para posterior análise. Constatou-se que a maioria dos autores formula o problema de sequenciamento para filas de espera fazendo uso de modelos com n variáveis quando existem n classes de clientes, ou seja, n buffers. Do ponto de vista desta tese, tal formulação não é a mais correcta. Colocou-se como hipótese que para estes modelos representarem correctamente os sistemas deverão ser modelados usando $n + k$ variáveis e não n. As k variáveis adicionais deverão reflectir o estado dos k servidores existentes na rede, sendo possível saber se o servidor se encontra livre ou que classe de clientes está a processar.

De modo a ser determinado se este tipo de modelos corresponde mais fielmente às redes de filas de espera reais, procedeu-se ao cálculo de custos de acordo com os dois tipos de modelos, através da formulação de um problema de programação dinâmica e respectiva solução por aplicação do algoritmo de iteração de valor. Posteriormente compararam-se estes resultados com o que se obtém através de um simulador de eventos discretos, onde é aplicada a política óptima para o problema clássico de sequenciamento para servidor único. O modelo correcto deverá conduzir a custos de estado semelhantes aos custos de estados obtidos através da simulação da rede em causa.

1.1 Estrutura da dissertação

Esta tese encontra-se dividida em seis capítulos e um anexo. No presente capítulo, pretende-se fazer um breve exposição dos principais temas desenvolvidos, bem como, fazer uma ligeira revisão dos conteúdos de cada capítulo.

O segundo capítulo, chamado Enquadramento, contém uma revisão da literatura dos pontos relevantes abordados nesta tese e faz referência a alguma ferramentas usadas na resolução dos problemas formulados. As secções existentes apresentam alguma teoria relativa a redes de filas de espera, cadeias de Markov, programação dinâmica e políticas de sequenciamento.

No terceiro capítulo, é feita uma exposição dos problemas em estudo assim como uma formulação teórica dos mesmos. Na primeira secção, é abordado o problema da modelação de redes de filas de espera, tendo como exemplo fila de servidor único, segundo duas vertentes. Na segunda secção, é formulado um problema de políticas de sequenciamento que fazem uso de prioridades.

O quarto capítulo apresenta o simulador. O simulador corresponde a um pacote de simulação de eventos discretos desenvolvido em JAVA, propositadamente para a simulação de redes de filas de espera ou de redes de actividades. A primeira secção, apresenta as características gerais do simulador e as seguintes fazem uma descrição mais detalhada das componentes do simulador bem como uma exemplificação de possível utilização do mesmo.

No capítulo cinco é feita uma exposição dos resultados derivados das soluções implementadas para os problemas formulados no capítulo três. Existem duas secções principais que reportam aos resultados dos dois problemas formulados, encontrando-se cada uma dessas secções divididas em em outras subsecções onde são apresentados os resultados da programação dinâmica da simulação e é feita um discussão dos mesmos.

Finalmente, o último capítulo apresenta algumas conclusões do trabalho realizado bem como alguma possibilidade de desenvolvimento futuro do trabalho realizado.

O anexo apresenta o *Unified Model Language* do simulador de eventos discretos.

Capítulo 2

Enquadramento

2.1 Redes de Filas de Espera

As redes de filas de espera são um tipo particular de sistemas de eventos discretos e, no caso concreto desta tese, o principal objecto de estudo consiste no problema de sequenciamento para filas de espera com servidor único.

As redes de filas de espera são sistemas dinâmicos constituídos por entidades processadoras pelas quais fluem clientes que aguardam a sua vez em fila. Depois de servidos numa dada entidade, os clientes deslocam-se para uma nova fila de espera ou abandonam o sistema.

2.1.1 Modelos de Filas de Espera

As redes de filas de espera são constituídas por três elementos: clientes ou materiais, servidores ou máquinas e buffers. No caso de ser estar perante um sistema de manufactura que represente, por exemplo, uma fábrica que seja modelada por uma rede de filas de espera, os servidores serão usualmente chamados máquinas e os clientes materiais. No seguimento deste texto, usar-se-ão apenas os termos cliente(s) e servidor(es). É ainda de ter em conta que um sistema que seja constituído por apenas um buffer e um servidor designa-se fila de espera, do qual é exemplo a Figura 2.1.

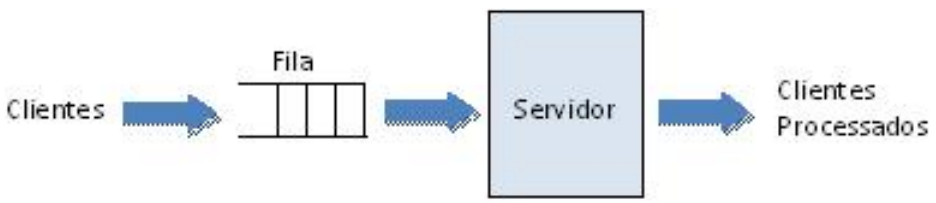


Figura 2.1: Exemplo de uma rede de filas de espera com um servidor e uma classe de clientes.

Para que seja possível caracterizar totalmente uma rede de filas de espera, é necessário saber-se mais do que por quantos elementos de cada tipo a mesma é constituída. A parte mais relevante para a caracterização de uma rede de filas de espera passa pela descrição da dinâmica entre os elementos que a constituem. Finalmente, sabendo o número de elementos de cada tipo que constituem a rede e o modo como os mesmos interagem, pode construir-se um modelo para uma dada rede de filas de espera.

A dinâmica das redes de filas de espera pode ser usualmente caracterizada por três componentes:

- Descrição dos processos de chegada à rede;
- Especificação dos parâmetros estruturais da rede, como seja o número de servidores e topologia;
- Especificação das políticas de operação usadas como seja a admissão de clientes à rede ou o modo como é dada prioridade aos diferentes clientes.

Reportando aos sistemas de eventos discretos, pode dizer-se que no caso de uma rede de filas de espera simples, o conjunto de eventos, E, será constituído por apenas dois eventos, $E = \{a, d\}$, onde a corresponde ao evento chegada de um cliente e d ao fim de serviço de um cliente e partida do sistema.

2.1.2 Modelos para as Chegadas e Processamentos

Os modelos das chegadas e processamentos podem ser de dois tipos: determinísticos ou estocásticos. Contudo, os modelos determinísticos têm pouco interesse em geral, sendo habitualmente usados modelos estocásticos. De acordo com [8],

Definição 1 *Um processo estocástico $X(\omega, t)$ é uma colecção de variáveis aleatórias indexadas por t. As variáveis aleatórias são definidas num conjunto (Ω, E, P) com $\omega \in \Omega$. A variável t pertence ao intervalo $T \subseteq \Re$.*

Na definição acima, E é um conjunto de eventos, Ω o conjunto de espaço de estados e P uma medida de probabilidade. O processo pode ainda ser caracterizado de acordo com o tempo. Tem-se um processo estocástico em tempo contínuo no caso de t poder tomar valores num conjunto real ou um processo estocástico em tempo discreto no caso de t assumir valores inteiros num conjunto contável. Se a variável, $X(t)$, puder tomar valores num intervalo contável, está-se perante um processo estocástico de variável discreta. Caso contrário diz-se que o processo é de variável contínua.

Voltando agora aos tempos de chegada e de processamento e ao caso concreto desta tese, serão usados processos de tempo contínuo e de estado discreto. Mais concretamente, os processos de

chegada são modelados por uma distribuição *Poisson* e os tempos de processamento distribuídos exponencialmente.

2.1.3 Parâmetros Estruturais

Tal como referido anteriormente, o número de servidores e o número de buffers permitem fazer a construção física da rede. Contudo, há ainda outros aspectos a ter em conta relativamente a estes constituintes: o número de clientes que um buffer pode conter, o número máximo de clientes que podem chegar à rede e o número de clientes na rede.

No caso dos buffers, dois cenários podem ter-se em conta: ou o buffer pode admitir um número limitado de clientes ou pode ter capacidade infinita. No que respeita ao número máximo de clientes que a rede pode admitir, tal como no caso dos buffers, a rede poderá admitir um número finito de clientes ou então aceitar um número ilimitado de clientes. Pode ainda ter-se a situação em que o número total de clientes que acede à rede constituir um população de tamanho finito ou ser ilimitada. Quando uma rede possuir um número constante de clientes no seu interior tem-se uma rede de filas de espera fechada. No caso de poder admitir um número variável de clientes tem-se uma rede de filas de espera aberta. No caso do presente trabalho, considera-se que os buffers têm capacidade infinita e que podem chegar à rede de filas de espera um número ilimitado de clientes, retirados de uma população ilimitada.

No que diz respeito à topologia, esta faz referência ao fluxo dos clientes pela rede bem como ao modo como os servidores e os buffers estão interligados.

2.1.4 Políticas de Operação

Para que seja possível falar acerca de políticas de operação de uma rede de filas de espera, é necessário, primeiramente, classificar a rede de filas de espera no que concerne às classes de clientes que possam existir, bem como o tipo de processamento que os servidores podem efectuar.

Quando se torna necessário distinguir os clientes que chegam ao sistema, estes são agrupados em classes de acordo com o buffer de entrada e o seu percurso ao longo da rede. Podem então existir sistemas de filas de espera multiclasse ou uniclasse dependendo se existe na rede mais de uma classe ou apenas uma classe de clientes.

A figura 2.1 apresenta um exemplo de um sistema uniclasse, onde os clientes independentemente do seu passado são indistinguíveis entre si sendo todos processados do mesmo modo.

Numa rede multiclasse, existe a possibilidade de o servidor distinguir os clientes de diferentes classes para posteriormente os processar de diferente modo se for caso disso.

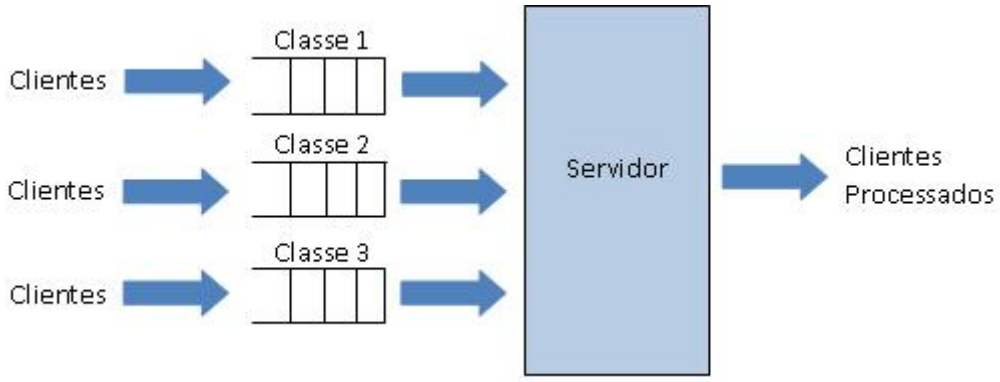


Figura 2.2: Rede de filas de espera com três classes de clientes.

A figura 2.2 representa uma rede de filas de espera constituída por um servidor e três classes de clientes. Os clientes são então distinguidos pela classe a que pertencem de acordo com o buffer em que estão inseridos.

Tendo ainda em conta a política de operação pela qual o servidor escolhe o tipo de cliente, podem podem ainda considerar-se dois modos de serviço possíveis. Voltando ao exemplo da figura 2.2 onde existem três tipos de clientes, podemos assumir que o servidor realiza um serviço do tipo *non-preemptive*. Ou seja, uma vez que um serviço é começado este deve ser terminado não havendo interrupções do mesmo para começar a servir outro cliente que possa ter entretanto chegado ao sistema, e que pertença a uma classe mais prioritária. No caso contrário ao anterior, diz-se que o serviço é do tipo *preemptive*.

Quanto às políticas de operação, existem três grandes grupos: políticas de sequenciamento, políticas de encaminhamento e políticas de admissão. As políticas de admissão têm como objectivo controlar a entrada de clientes na rede, decidindo se um cliente pode ser alojado na rede ou se é rejeitado. Através do encaminhamento é possível determinar-se correctamente em que buffer deve ser colocado um cliente das várias possibilidades que se possam apresentar. Finalmente, no que diz respeito às políticas de sequenciamento, estas podem ser tidas em conta como duais das políticas de encaminhamento uma vez que, enquanto que no encaminhamento se decide para que buffer os clientes devem ser enviados, no caso do sequenciamento importa decidir de que buffer devem ser retirados os clientes para serem processados. Estas políticas são portanto centradas no servidor que deverá escolher de que classe das existentes deverá processar. Na Figura 2.3 pode ver-se o esquema de aplicação de políticas de sequenciamento ao sistema da figura 2.2.

É de fácil conclusão que as políticas de sequenciamento só se justificam quando exista mais de um buffer de onde um servidor possa retirar clientes.

Ainda no que diz respeito às políticas de sequenciamento estas podem ser classificadas quanto

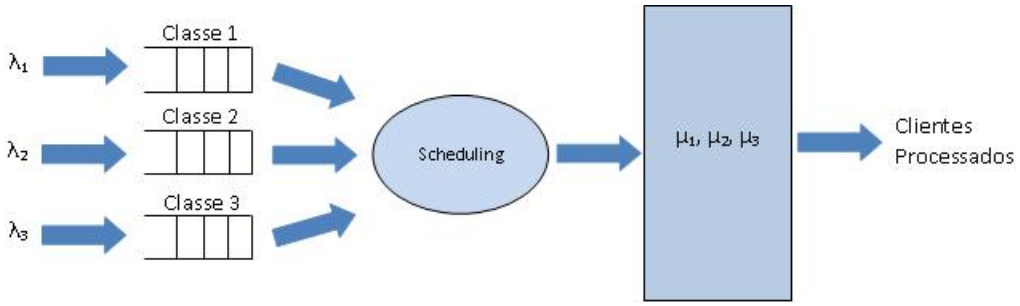


Figura 2.3: Sequenciamento de três classes de clientes para um servidor.

à sua localidade e *idleness* (capacidade de os servidores poderem estar parados mesmo existindo clientes à espera para serem processados):

- no caso da política ser local, a decisão é baseada apenas em informação local, ou seja, informação relativa aos clientes que aguardam para ser processadas pelo servidor que executa a política;
- no caso da política não ser local, o servidor usa informação contida em toda a rede;
- numa política *idling*, cada servidor tem a possibilidade de não servir ninguém mesmo quando existem clientes à espera de serem processados pelo mesmo;
- no caso da política de sequenciamento ser *non-idling*, o servidor estará sempre a processar enquanto existirem clientes.

O presente trabalho pretende contribuir na área das políticas de sequenciamento, não fazendo neste caso sentido falar-se de encaminhamento uma vez que se tem um sistema de servidor único. No que diz respeito à admissão, também não será usado nenhum algoritmo de controlo.

2.1.5 Estabilidade das Redes de Filas de Espera

Quando se está perante uma rede de filas de espera, é desejável que os clientes estejam o mínimo de tempo possível à espera. Por outro lado, é também desejável que seja possível processar todos os clientes que chegam ao sistema num tempo considerado razoável. Uma forma de garantir que as características anteriormente citadas se verifiquem prende-se com o facto de a rede ser estável.

Conjectura-se então, que a estabilidade da rede está directamente relacionada com a intensidade de tráfego, ou seja, com o facto de cada servidor de rede ter recursos suficientes para processar todos os clientes que lhe são encaminhados.

Definição 2 *Considere-se uma rede de filas de espera composta por I servidores onde μ_i^k é o tempo médio de processamento da classe k pelo servidor i e λ_i^k é a média da taxa de chegada da classe k ao*

servidor i. A Intensidade de Tráfego no servidor i vem então dada pela seguinte expressão:

$$\rho_i = \sum_{k=1}^{N_i} \mu_i^k \lambda_i^k c\left(k,i\right), \tag{2.1}$$

onde

$$c\left(k,i\right) = \begin{cases} 1 \;\; \textit{se a classe } k \;\; \textit{é servida pelo servidor } i \\ 0 \;\; \textit{caso contrário} \end{cases}$$

A Condição de Intensidade de Tráfego requer para cada servidor $i = 1, ..., I$ que $\rho_i \leq 1$. A condição anterior é necessária para garantir a estabilidade de uma rede de filas de espera. Esta condição pode facilmente ser verificada intuitivamente uma vez que tendo $\rho_i \leq 1$ a taxa de chegada de clientes será menor do que a respectiva taxa de processamento sendo o servidor capaz de processar um ou mais clientes existentes na rede antes que um outro chegue. Por outro lado, se $\rho_i > 1$ é óbvio que o servidor não será capaz de processar os clientes tão rápido quanto eles chegam à rede, potenciando a criação de longas filas de espera.

2.1.6 Sequenciamento

Sendo o problema de sequenciamento em servidor único o principal objecto de estudo desta tese, far-se-á nesta secção uma análise continuada à feita na secção 2.1.4 acerca deste tema bem como uma abordagem à política de sequenciamento μc-rule.

Tal como referido anteriormente, o sequenciamento tem como objectivo escolher a classe de clientes a processar de entre as classes de clientes existentes na rede. Esta escolha pode ser feita de acordo com diversos objectivos como sejam a redução do tempo total de espera, redução dos desvios em relação a datas de entrega, etc.. Assim, os clientes podem ser escolhidos com base na ordem de chegada ao sistema, *first-come-first-served*, com base em datas de entrega, por prioridade a classes, etc...

Um típico objectivo do sequenciamento é minimizar o custo total esperado descontado numa rede de filas de espera, [8]. Considere-se o exemplo da Figura 2.4.

Neste caso, a rede é constituída por duas classes de clientes onde é aplicado um algoritmo de sequenciamento. Assuma-se que que cada uma das classes de clientes tem um parâmetro de custo indexado $c_i, i = 1, 2$, que traduz o custo associado a um cliente por unidade de tempo, sendo por norma $c_1 \neq c_2$. Considere-se ainda que cada classe de clientes é processada a uma taxa $\mu_i, i = 1, 2$, sendo por norma $\mu_1 \neq \mu_2$. O custo total esperado descontado, para horizonte infinito, expressa-se como

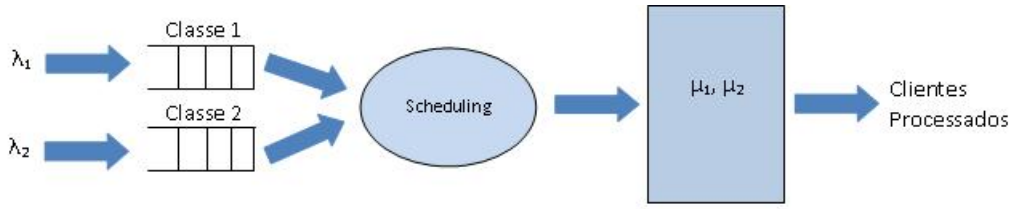


Figura 2.4: Exemplo de controlo de sequenciamento aplicado a uma rede com duas classes de clientes.

$$V_\pi = (x_1(0), x_2(0)) = \underset{\pi}{E}\left[\int_0^\infty e^{-\beta t}\left\{c_1 x_1(t) + c_2 x_2(t)\right\} dt\right], \tag{2.2}$$

onde β é um factor de desconto, $\{x_1(0), x_2(0)\}$ é o estado inicial e $x_i(t)$ o tamanho do buffer i no instante t, para $i = 1, 2$. De modo a que seja possível a formulação de um problema de Markov - ver secção 2.2 -, de acordo com as secções anteriores, considera-se ainda que os clientes chegam aos buffers de acordo com processo de *Poisson* com taxa média $\lambda_i, i = 1, 2$ e que os tempos de serviço são distribuídos exponencialmente com taxa média $\mu_i, i = 1, 2$.

De acordo com [8], pode mostrar-se que a política de sequenciamento óptima a ser aplicada num caso como o exemplo acima é a μc-rule, ou seja, a prioridade de serviço é sempre dada à classe não vazia com o maior valor de $\mu_i c_i$. A prova de optimalidade desta regra é feita considerando vários tipos de políticas e calculando o seu respectivo custo obtendo-se por fim que a política que minimiza o custo é a μc-rule. A prova intuitiva pode ser consultada em [8].

No que diz respeito à criação desta regra, foi formulada por Cox [10] no anos 60 de acordo com uma perspectiva probabilística. Posteriormente, outros autores em [2] fizeram uma outra abordagem da mesma política com base em técnicas de programação dinâmica. Para consultar a demonstração da optimalidade desta política ver [7, 2].

2.2 Cadeias de Markov

Uma rede de filas de espera pode ser descrita por um processo de Markov, quando determinadas propriedades se verificam. A partir da formulação de um Problema de Decisão de Markov, bastantes características decorrentes da geração do mesmo podem ser de grande utilidade.

Primeiramente, importa referir que uma cadeia de Markov é, na sua essência, um processo estocástico. Sendo assim, podem existir à semelhança do que acontece com qualquer processo estocástico como referido na secção 2.1.2, cadeias de Markov em tempo contínuo e em tempo discreto bem como cadeias de Markov de estado contínuo e de estado discreto. No caso do presente trabalho, será dada atenção às cadeias de Markov em tempo contínuo e discreto mas apenas de estado discreto.

Segundo Walrand [21], a definição de cadeia de Markov é a seguinte: uma cadeia de Markov $x = \{x_t, t \geq 0\}$ descreve a evolução aleatória de uma partícula "amnésica" num conjunto contável. A definição anterior equivale a dizer que o processo x esqueceu o modo pelo qual chegou ao presente estado x_t, e a chegada aos estados seguintes também será feita sem recorrer aos estados anteriores. Em suma, a partícula assume sempre que parte de $t = 0$.

2.2.1 Cadeias de Markov em Tempo Discreto e Estado Discreto

Considere-se um processo estocástico $\{X_n, n = 0, 1, 2, ...\}$ que toma valores num conjunto finito, de cardinalidade B. Sendo $X_n = i$, diz-se que o processo se encontra no estado i no momento n. Partindo deste processo, é possível formar uma sequência $\{X_1, X_2, ...\}$ caracterizada pela Propriedade de Markov [20]:

$$P\{X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, ..., X_1 = i_1, X_0 = i_0\} = P\{X_{n+1} = j | X_n = i\}. \tag{2.3}$$

Supõe-se ainda que sempre que o processo se encontra no estado i, existe uma probabilidade p_{ij} de que o estado seguinte seja o estado j. Posto isto, tem-se:

$$P\{X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, ..., X_1 = i_1, X_0 = i_0\} = p_{ij}, \tag{2.4}$$

para qualquer estado $i_0, i_1, ..., i_{n-1}, i_n, j$ para $n \geq 0$. O processo considerado é então chamado de cadeia de Markov, sendo a distribuição condicional de qualquer estado futuro, X_{n+1}, dados os estados passados, $\{X_0, X_1, ..., X_{n-1}\}$, e o estado presente, X_n, independente desses mesmos estados passados e dependente apenas do estado presente.

O valor da probabilidade p_{ij} representa a probabilidade de o processo transitar do estado i para o estado j. Uma vez que estas probabilidades são sempre não negativas, e que que deverá haver sempre uma transição de estado tem-se que:

$$\sum_{j=0}^{B-1} p_{ij} = 1, i = 0, 1, \ldots, B-1. \tag{2.5}$$

Para uma cadeia de Markov genérica pode portanto obter-se a matriz de transição de estado P (*matrix of one-step transitions probabilities*):

$$P = \begin{bmatrix} p_{00} & p_{01} & p_{02} & \cdots \\ p_{10} & p_{11} & p_{12} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ p_{i0} & p_{i1} & p_{i2} & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{2.6}$$

Se o número de estados for ilimitado diz-se que se tem um kernel de probabilidades de transição.

Definidas as transições *one-step* de probabilidade, p_{ij}, podem ainda definir-se as transições de probabilidade a n passos, p_{ij}^n. Ou seja, a probabilidade de o processo no estado i passar para o estado j após n transições é dada pela seguinte expressão:

$$p_{ij}^n = P\left\{X_{n+m} = j | X_m = i\right\}, n \geq 0, i, j \geq 0. \tag{2.7}$$

Como é de fácil observação, $p_{ij}^1 = p_{ij}$. Estas probabilidades de transição podem ser calculadas fazendo uso das equações de Chapman-Kolmogorov, definidas como:

$$p_{ij}^{n+m} = \sum_{k=0}^{\infty} p_{ik}^n p_{kj}^m, n, m \geq 0, i, j \geq 0. \tag{2.8}$$

2.2.2 Cadeias de Markov em Tempo Contínuo e Estado Discreto

Para caracterizar este tipo de cadeias, será feita uma descrição em concordância com o que foi feito na secção anterior para as cadeias em tempo discreto. Considere-se um processo estocástico em tempo contínuo $\{X(t), t \geq 0\}$. Diz-se que o processo é uma cadeia de Markov em tempo contínuo se para todo $s, t \geq 0$ e para valores inteiros $i, j, x(u), 0 \leq u < s$ se tem:

$$P\left\{X(t+s) = j | X(s) = i, X(u) = x(u), 0 < u < s\right\} = P\left\{X(t+s) = j | X(s) = i\right\}. \tag{2.9}$$

A equação anterior equivale a dizer que uma cadeia de Markov em tempo contínuo é um processo estocástico onde a Propriedade Markoviana impõe que a distribuição do estado futuro no momento $t+s$ dado o estado presente t dependa apenas do estado presente independentemente dos estados passados. Adicionalmente, tem-se que se $P\{X(t+s)=j|X(s)=i\}$ é independente de s, diz-se que a cadeia de Markov em tempo contínuo tem transições de probabilidade estacionárias.

Suponha-se que uma cadeia de Markov em tempo contínuo entra no estado i num dado instante de tempo e que o processo não sai desse mesmo estado i, ou seja, que a transição não ocorre durante um intervalo de duração s. Qual é a probabilidade de o processo não deixar o estado i durante um intervalo de tempo posterior de tamanho t? Para responder a esta pergunta deve primeiramente ter-se em conta que o processo se encontra no estado i no instante de tempo s. Portanto, pela propriedade Markoviana, a probabilidade de o processo permanecer no estado i durante o intervalo de tempo $[s, s+t]$ é a probabilidade de o processo permanecer no estado i por pelo menos um intervalo de tempo t. Sendo τ_i o tempo que o processo permanece no estado i antes de fazer a transição para um estado diferente, tem-se que:

$$P\{\tau_i > s+t|\tau_i > s\} = P\{\tau_i > t\}, \tag{2.10}$$

para todo $s, t \geq 0$. A variável aleatória τ_i é então desprovida de memória sendo condição necessária e suficiente que esta seja exponencialmente distribuída. Dadas a condições acima, é possível construir uma cadeia de Markov em tempo contínuo. Pode acrescentar-se ainda que se trata de um processo estocástico tendo as seguintes propriedades a cada instante em que entra no estado i:

- a quantidade de tempo que é gasta no estado antes de ser feita uma transição para um diferente estado é exponencialmente distribuída com taxa γ_i;
- quando o processo abandona o estado i este passará ao estado j com uma probabilidade p_{ij} sendo $\sum_{ij} p_{ij} = 1$.

Em forma de conclusão, pode então dizer-se de uma cadeia de Markov em tempo contínuo que é um processo estocástico que transita de estado para estado em concordância com uma cadeia de Markov em tempo discreto, mas na qual o tempo despendido em cada estado antes de proceder à transição seguinte é exponencialmente distribuído. Adicionalmente, a quantidade de tempo que o processo despende no estado i e o próximo estado a ser visitado são variáveis aleatórias independentes. Caso contrário, a assumpção Markoviana seria contrariada. Seja $q_{ij} \geq 0$ definido por:

$$q_{ij} = \lim_{t \to \infty} \frac{P\{X(t+s) = j|X(s) = i\}}{t}. \tag{2.11}$$

A quantidade q_{ij} representa a taxa à qual o processo no estado i transita para o estado j, com $j \neq i$, sendo portanto q_{ij} chamada a taxa de transição do estado i para o estado j. Define-se ainda $p_{ij}(t)$ como sendo a probabilidade de que a cadeia de Markov presentemente no estado i estará no estado j após um intervalo de tempo t, ou seja:

$$p_{ij}(t) = P\{X(t+s) = j | X(s) = i\}. \tag{2.12}$$

À semelhança do que foi feito com as cadeias de Markov em tempo discreto pode então obter-se uma matriz de transição Q que mostra as taxas de transição de estado para estado:

$$Q = \begin{bmatrix} -q_{00} & q_{01} & q_{02} & \cdots \\ q_{10} & -q_{11} & q_{12} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ q_{i0} & q_{i1} & \cdots & -q_{ii} \end{bmatrix} \tag{2.13}$$

Através da análise da matriz Q pode então tirar-se a seguinte conclusão: os valores da diagonal da matriz, $-q_{ii} = -\sum_j q_{ij}$, representam a taxa total de transição do evento i. Portanto, a soma dos elementos de uma linha desta mesma matriz é igual a zero.

2.2.3 Uniformização

As cadeias de Markov em tempo contínuo possibilitam, tal como visto anteriormente, que as transições ou eventos aconteçam em qualquer instante. Esta propriedade, tem como consequência que estas cadeias correspondam mais fielmente à evolução dos processos reais do que as cadeias em tempo discreto. Contudo, verifica-se geralmente que o uso de cadeias de Markov em tempo discreto é mais conveniente e torna o problema mais fácil de analisar. A solução é então a conversão de modelos em tempo contínuo para modelos em tempo discreto isto sem que a informação inicialmente contida no modelo em tempo contínuo seja distorcida. A esta conversão dá-se o nome de Uniformização de Cadeias de Markov em Tempo Contínuo ou *Uniformização de Lipman* [17].

Dada a cadeia de Markov em tempo contínuo com espaço de estados E e matriz de taxas de transição Q o processo de uniformização consiste na construção de uma cadeia de Markov em tempo discreto uniformizada estocasticamente equivalente, calculando uma única taxa de transição:

$$\gamma \geq \max_{i \in E} \{q_{ii}\} \tag{2.14}$$

e probabilidade de transição

$$p_{ij} = \begin{cases} \dfrac{q_{ij}}{\gamma} & se \quad i \neq j \\ 1 - \dfrac{q_{ii}}{\gamma} & se \quad i = j \end{cases}$$

2.2.4 Problemas de Decisão de Markov

As cadeias de Markov consideradas nas secções anteriores têm todas probabilidades de transição fixas. Nesta secção pretende-se discutir a possibilidade de adicionar acções de controlo aos sistemas de modo a que a evolução da cadeia seja afectada. Usualmente, é pretendido que as cadeias sejam controladas para que algum tipo de comportamento seja atingido. Para que tal se possa verificar é necessário formular critérios de optimalidade bem como integrá-los em problemas de controlo.

Considere-se um Sistema de Eventos Discretos (DES) com espaço de estados E onde todas as transições de estado podem ser observadas. De modo a que se possam relacionar DES com problemas de decisão de Markov, (PDM), três aspectos têm de ser tidos em conta: acções de controlo, tomadas no instante em que uma transição de estado acontece, o custo associado a cada transição de estado e/ou ao tempo de permanência no estado e à forma como as transições dependem do controlo.

Quando se chega a um novo estado, uma acção de controlo u é seleccionada de um conjunto conhecido de possíveis acções de controlo U. Associado à selecção dessa mesma acção de controlo u no estado $i \in E$ estará um custo $C\left(i,u\right)$ que se assume ser não negativo e limitado, $0 \leq C\left(i,u\right) \leq K$. Por fim, é possível definir claramente problemas de optimização possibilitando a sua solução através de metodologias existentes.

A regra segundo a qual as acções de controlo são tomadas é chamada política, π. A política pode, por exemplo, ser arbitrária de modo a possibilitar a selecção aleatória de uma acção de controlo do conjunto de acções possíveis ou então ter em atenção informações do problema. No caso particular desta tese tem interesse estudar políticas que obedeçam a dois critérios: que não escolham a acção de controlo de modo aleatório e que escolham a acção a ser tomada no estado i apenas de acordo com informação desse mesmo estado, isto é, políticas estacionárias.

Segundo políticas estacionárias, as acções de controlo são mapeadas entre o conjunto de estados possíveis E e o conjunto de acções possíveis U existindo funções tipo $u\left(i\right)$, $i \in E$. Não sendo por vezes possível realizar todas as acções contidas em U, é ainda possível admitir a existência de um outro conjunto de acções U_i, que corresponde às acções possíveis de serem executadas no estado i.

Escolhida a acção a ser tomada no estado i, a transição para o estado seguinte far-se-á de acordo com as probabilidades de transição $p_{ij}\left[u\left(i\right)\right]$, que dependem apenas do valor de i, depois de se ter associado com o estado i a decisão $u(i)$. Assume-se ainda que os estados têm tempos de latência

exponencialmente distribuídos no tempo com taxa média $\Lambda(i,u)$ correspondendo estas especificações à descrição de uma cadeia de Markov onde a novidade é o facto de as probabilidades de transição de estado $p_{ij}[u(i)]$ dependerem da política π que se adoptar.

Deve ter-se em conta que o estado de uma cadeia de Markov contável pode ser representado por mais do que um simples inteiro existindo casos em que o estado é representado usando por exemplo vectores ou mesmo matrizes. Pode então dizer-se que a probabilidade de transição de um estado s para um estado s' de acordo com uma acção de controlo u é da forma $p_{ss'}[u(s)]$, onde s e s' são representados da forma mais conveniente para um dado problema.

O objectivo deste tipo de problemas passa então por obter a política óptima para um dado problema de acordo com algum critério de custo. Considere-se o processo $\{X(t)\}$. Se no instante de tempo t o estado for dado por $X(t)$, uma política π for especificada e uma determinada acção $u(t)$ for tomada unicamente de acordo com $X(t)$, deverá existir um custo $C[X(t), u(t)]$. Existem várias formas de contabilizar os custos incorridos durante um determinado horizonte de controlo. Sobre esta questão ver Secção 2.3.

No caso concreto desta tese será usado um critério de custo baseado no custo total descontado esperado num horizonte infinito. Este desconto é realizado através de um factor de desconto $\beta \geq 0$ permitindo assim a acumulação de custos descontados durante um futuro infinito e também garantindo a convergência do custo para valores finitos em horizonte infinito:

$$V_\pi(x_0) = E_\pi\left[\int_0^\infty e^{-\beta t} C[X(t), u(t)]\, dt\right]. \tag{2.15}$$

O sub-índice π significa que o custo resulta de uma trajectória obtida através da aplicação da política π durante o horizonte de controlo. O valor esperado é tomado sobre todas as trajectórias possíveis para a cadeia de Markov sob a política π. É graças ao factor exponencial que o custo é descontado. O custo é sempre definido como função do estado inicial, x_0, que se assume ser conhecido.

Finalmente, e uma vez que o critério de custo foi definido em tempo contínuo, resta então perceber o que acontece a este critério quando se faz a uniformização de um processo de decisão de Markov.

De acordo com os conteúdos da secção anterior, considere-se a taxa de transição única γ e que os instantes de transição ocorrem nos instantes $T_o, T_1, ..., T_k, ...$ sendo $T_0 = 0$. É então possível reescrever o critério de custo da seguinte forma:

$$E_\pi\left[\int_0^\infty e^{-\beta t} C[X(t), u(t)]\, dt\right] = \sum_{k=0}^{\infty} E_\pi\left[\int_{T_k}^{T_{k+1}} e^{-\beta t} C[X(t), u(t)]\, dt\right]. \tag{2.16}$$

Contudo é de ter em conta que o custo se mantém constante no intervalo $[T_{k+1}, T_k[$ sendo possível substituir $C\left[X\left(t\right), u\left(t\right)\right]$ para $T_k \leq t < T_{k+1}$ por um custo $C\left(X_k, u_k\right)$ sendo este actualizado assim que a k-ésima transição ocorra. Para além destes factos o custo depende apenas da politica π e do estado X_k, portanto, a equação 2.16 pode ser reescrita da seguinte forma:

$$E_\pi\left[\int_0^\infty e^{-\beta t} C\left[X\left(t\right), u\left(t\right)\right] dt\right] = \sum_{k=0}^{\infty} E_\pi\left[\int_{T_k}^{T_{k+1}} e^{-\beta t} dt\right] E_\pi\left[C\left(X_k, u_k\right)\right]. \tag{2.17}$$

Considere-se o caso em que $\beta > 0$, o lado direito da equação 2.17 vem dado por:

$$E_\pi\left[\int_{T_k}^{T_{k+1}} e^{-\beta t} dt\right] = -\frac{1}{\beta}\left[E_\pi\left[e^{-\beta T_{k+1}}\right] - E_\pi\left[e^{-\beta T_k}\right]\right], \tag{2.18}$$

sendo $T_{k+1} = T_k + Q_{k+1}$, onde Q_{k+1} corresponde ao tempo de permanência no estado após a k-ésima transição:

$$\begin{aligned} E_\pi\left[\int_{T_k}^{T_{k+1}} e^{-\beta t} dt\right] &= -\frac{1}{\beta}\left[E_\pi\left[e^{-\beta(T_k+Q_{k+1}}\right] - E_\pi\left[e^{-\beta T_k}\right]\right] \\ &= \frac{1}{\beta} E_\pi\left[e^{-\beta T_k}\right]\left[1 - E_\pi\left[e^{-\beta Q_{k+1}}\right]\right], \end{aligned} \tag{2.19}$$

onde foi usado o facto de Q_{k+1} ser independente de T_k. Como Q_{k+1} é distribuído exponencialmente de acordo com o parâmetro γ, tem-se

$$E_\pi\left[e^{-\beta Q_{k+1}}\right] = \int_0^\infty e^{-\beta t} \gamma e^{-\gamma t} dt = \frac{\gamma}{\beta + \gamma}. \tag{2.20}$$

Como $T_k = Q_1 + ... + Q_k$, onde todos os tempos de permanência são mutuamente independentes, obtém-se

$$E_\pi\left[e^{-\beta T_k}\right] = E_\pi\left[e^{-\beta Q_1}\right] ... E_\pi\left[e^{-\beta Q_k}\right] = \left(\frac{\gamma}{\beta + \gamma}\right)^k. \tag{2.21}$$

Defenindo $\alpha = \frac{\gamma}{\beta+\gamma}$ obtém-se:

$$E_\pi\left[\int_0^\infty e^{-\beta t} C\left[X\left(t\right), u\left(t\right)\right] dt\right] = \sum_{k=0}^{\infty} \frac{\alpha^k (1-\alpha)}{\beta} E_\pi\left[C\left(X_k, u_k\right)\right]. \tag{2.22}$$

Finalmente, sendo $(1 - \alpha) = \beta/(\beta + \gamma)$ o critério de custo vem dado por:

$$E_\pi\left[\int_0^\infty e^{-\beta t} C\left[X\left(t\right), u\left(t\right)\right] dt\right] = \frac{1}{\beta + \gamma} E_\pi\left[\sum_{k=0}^{\infty} \alpha^k C\left(X_k, u_k\right)\right]. \tag{2.23}$$

Para $\beta = 0$, é também possível mostrar que se obtém a mesma estrutura sendo a sua derivação omitida neste texto (ver [8]). Em suma, o problema de determinação da política π para minimizar um critério de custo 2.15 para uma cadeia de tempo contínuo é convertido num problema para determinar uma política π para minimizar 2.23 dado:

- uma cadeia de Markov em tempo discreto com uma taxa de transição única γ;
- um factor de desconto $\frac{\gamma}{\beta+\gamma}$ com $0 < \alpha < 1$;
- um custo dado por $\frac{C(i,u)}{(\beta+\gamma)}$ quando no estado i é escolhida a acção de controlo u.

2.3 Programação Dinâmica

O início da Programação Dinâmica, (PD), data dos anos 50, aquando da publicação dos primeiros artigos de Bellman, [3, 4], que introduziam esta mesma teoria. Desde então, a PD tem vindo a tornar-se uma disciplina de interesse maior em áreas como a matemática aplicada e a investigação operacional bem como uma ferramenta standard de outras áreas como seja a engenharia, a economia, a gestão entre muitas outras. Neste capítulo irão ser introduzidos alguns conceitos relacionados com o tema da PD.

Tal como referido, a PD pode ser usada para tratar uma grande variedade de problemas. No decorrer deste texto, iremos focar-nos apenas na formulação de modelos de controlo óptimo de sistemas dinâmicos. Dois tipos de problemas podem ter-se em conta, problemas de horizonte finito e problemas de horizonte infinito. No caso concreto desta tese, têm particular interesse os problemas de horizonte infinito.

2.3.1 Formulação de Problemas

O modelo a considerar para a formulação de um problema de PD [5], para um horizonte finito, deverá ter duas características principais: ser representativo de um sistema dinâmico, seja ele descrito por equações diferenciais, ou às diferenças ou ainda um sistema dinâmico de eventos discretos, e ter uma função de custo indexada que seja aditiva ao logo do tempo.

Considere-se o seguinte sistema

$$x_{k+1} = f_{k+1}\left(x_k, u_k, \omega_k\right), k = 0, 1, ..., N-1, \tag{2.24}$$

onde k indexa o tempo discreto, x_k é um elemento do conjunto S_k que representa o estado do sistema e concentra informação necessária para futura optimização, u_k representa a acção de controlo do conjunto C_k seleccionada no instante k, ω_k é um parâmetro aleatório do conjunto D_k e N representa o horizonte, ou seja, o número de vezes que o controlo é aplicado. A acção de controlo pode apenas tomar valores num conjunto não vazio $U\left(x_k\right) \subset C_k$ dependendo do estado corrente x_k, ou seja, $u_k \in U_k\left(x_k\right)$ para todo o $x_k \in S_k$. O parâmetro aleatório w_k é caracterizado por uma distribuição de probabilidade $P_k\left(\cdot|x_k, u_k\right)$ que pode depender explicitamente de x_k e u_k mas, em geral, não depende de perturbações anteriores $\omega_{k-1}, ..., \omega_0$.

É também considerada uma classes de políticas (ou leis de controlo) que consistem numa sequência de funções do tipo $\pi = \{\mu_0, ..., \mu_{N-1}\}$ onde μ_k faz correspondência com o estado x_k sendo o controlo

da forma $u_k = \mu_k(x_k)$ onde $\mu_k(x_k) \in U_k(x_k)$ para todo $x_k \in S_k$. Este tipo de políticas são chamadas de políticas admissíveis .

Dado um estado inicial x_0 e uma política admissível $\pi = \{\mu_0, ..., \mu_{N-1}\}$ a equação

$$x_{k+1} = f_k(x_k, \mu_k(x_k), \omega_k), k = 0, 1, ..., N-1, \tag{2.25}$$

faz com que as variáveis aleatórias x_k e ω_k, tenham distribuições bem definidas. Assim, para funções do tipo g_k , $k = 0, 1, ..., N-1$ o custo esperado

$$J_\pi(x_0) = \underset{\substack{\omega_k \\ k=0,1,...N-1}}{E} \left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), \omega_k\} \right), \tag{2.26}$$

é também uma quantidade bem definida. Tem-se então que para um dado estado inicial x_0 uma política óptima π^* é uma política que permite a minimização deste custo, ou seja,

$$J_{\pi^*}(x_0) = \min_{\pi \in \Pi} J_\pi(x_0), \tag{2.27}$$

onde Π é o conjunto de todas as políticas admissíveis. O custo óptimo depende então de x_0 e é denotado por $J^*(x_0)$, ou seja,

$$J^*(x_0) = \min_{\pi \in \Pi} J_\pi(x_0). \tag{2.28}$$

J^* é então tida como uma função que atribui a cada estado inicial x_0 o custo óptimo $J^*(x_0)$ sendo a mesma chamada de função de custo óptimo ou função de valor óptimo. No que diz respeito à expressão 2.28 deve ter-se em conta que $\min$ corresponde ao maior valor do limite inferior (ínfimo) do conjunto de valores $\{J_\pi(x_0) | \pi \in \Pi\}$.

2.3.2 O Algoritmo de Programação Dinâmica

A técnica da PD assenta numa simples ideia, o princípio de optimalidade. Tal como referido no início desta secção, este conceito deve-se a Bellman. O princípio de optimalidade pode ser então definido da seguinte forma:

Definição 3 *Seja $\pi^* = \{\mu_0^*, \mu_1^*, ..., \mu_{N-1}^*\}$ uma política óptima para o problema da subsecção anterior. Assuma-se ainda que quando π^* é usada um dado estado x_i ocorre no instante i com uma probabilidade positiva. Considere-se o subproblema onde estando no x_i no instante i se quer minimizar o custo*

entre o instante i e o instante N

$$E\left\{g_N\left(x_N\right)+\sum_{k=i}^{N-1} g_k\left(x_k,\mu_k\left(x_k\right),\omega_k\right)\right\}. \tag{2.29}$$

Conclui-se então que a política $\left\{\mu_0^*,\mu_1^*,...,\mu_{N-1}^*\right\}$ é óptima para este subproblema.

O princípio de optimalidade sugere então que um política óptima possa ser construída pedaço a pedaço, ou seja, primeiro é construída uma política óptima para os estados finais do problema, mais concretamente para a última transição, sendo construído o resto da política de acordo com este mesmo método.

Seguidamente, proceder-se-à à exposição do algoritmo de PD para um problema simples, sendo também mostrada a optimalidade através do uso de ferramentas matemáticas.

Proposição 1 *Proposição: Para todo o estado inicial x_0, o custo óptimo para um dado problema é dado por $J_0\left(x_0\right)$, onde a função de custo J_0 é dado pelo resultado do último passo do seguinte algoritmo, que se desenvolve do instante $N-1$ para o instante 0:*

$$J_N\left(x_N\right)=g_N\left(x_N\right), \tag{2.30}$$

$$J_k\left(x_k\right)=\min_{u_k\in U_k\left(x_k\right)} E_{\omega_k}\left\{g_k\left(x_k,u_k,\omega_k\right)+J_{k+1}\left(f_k\left(x_k,u_k,\omega_k\right)\right)\right\}, k=0,1,...,N-1, \tag{2.31}$$

onde o valor esperado é calculado relativamente à probabilidade de distribuição de ω_k, que por sua vez depende de x_k e u_k. Tem-se ainda que se $u_k^*=\mu_k^*\left(x_k\right)$ minimiza o lado direito da equação 2.31 para cada x_k e k, a política $\pi^*=\left\{\mu_0^*,\mu_1^*,...,\mu_{N-1}^*\right\}$ é óptima.

A demonstração desta proposição pode ser consultada em [5]. Ainda de acordo com a proposição anterior, $J_k\left(x_k\right)$ é chamado o custo para continuar a partir do estado x_k no instante k e J_k a função de custo no instante k.

Idealmente, o desejado seria, através da PD, obter expressões de forma fechada para J_k ou uma política óptima. Contudo, em muitos casos não é possível obter soluções analíticas optando-se por recorrer a métodos computacionais que executem o algoritmo de PD. Os detalhes mais técnicos relativos à formulação matemática deste tipo de problemas podem ser consultados em [5, 6].

2.3.3 Problema de Horizonte Infinito

Nesta secção algumas das propriedades dos problemas de Horizonte Infinito irão ser expostas. Quando comparados com os problemas de horizonte finito, duas diferentes características podem ser apontadas

aos problemas de horizonte infinito:

- o número de iterações é infinito;
- o sistema é estacionário, ou seja, a equação do sistema, o custo de estado e a estatística da perturbação não mudam de estado para estado.

Geralmente, os problemas de horizonte infinito, requerem uma análise mais sofisticada do que os problemas de horizonte finito. Decorrente desta sofisticação, por vezes esta análise pode não ser trivial.

Embora existam vários tipos de problemas de horizonte infinito, são de particular interesse, no âmbito desta tese, os problemas de minimização do custo descontados:

$$J_\pi(x_0) = \lim_{N\to\infty} \underset{\substack{\omega_k \\ k=0,1,\dots}}{E} \left\{ \sum_{k=0}^{N-1} \alpha^k g\left(x_k, \mu_k(x_k), \omega_k\right) \right\}, \tag{2.32}$$

onde $J_\pi(x_0)$ é o custo do estado inicial x_0, associado à política $\pi = \{\mu_0, \mu_1, ...\}$ e α é o factor de desconto, tal que $0 < \alpha \leq 1$.

Ainda no que diz respeito ao tema desta tese, importam ter em conta os problemas de custo de estado limitado, tendo-se que $|g(x, u, \omega)|$ é limitado superiormente por uma constante M. $J_\pi(x_0)$ corresponde à soma infinita de uma sequência de números que têm o seu valor absoluto limitado devido à progressão geométrica decrescente $\left\{\alpha^k M\right\}$.

2.3.4 Iteração de Valor

Para que fosse possível resolver o problema de minimização de custo exposto na secção anterior, optou-se por usar o algoritmo de Iteração de Valor.

Este algoritmo consiste na iteração do algoritmo de PD de acordo com a seguinte recursão:

$$J_{k+1}(i) = \min_{u\in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right], i = 1, ..., n. \tag{2.33}$$

Este método é um dos principais métodos usados na determinação de funções óptimas de custo J^*. Usualmente, este algoritmo requer um número infinito de iterações. Contudo, em alguns casos, o número de iterações advém finito, [6].

Capítulo 3

Formulação de Problemas

3.1 Modelação usando 1 Variável VS 2 Variáveis

O primeiro tema de estudo a ser desenvolvido nesta tese diz respeito ao número de parâmetros mínimos necessários para que seja reproduzida fielmente uma rede de filas de espera usando um modelo matemático. Ao consultar referências [12, 11, 9], relativas à formulação de diversos problemas relacionados com redes de filas de espera, mais concretamente com a obtenção de políticas de sequenciamento, verificou-se que a grande maioria dos autores, a não ser os autores que estudam políticas que usem um custo indexado à mudança de estado do servidor [15, 16], usam apenas a informação do conteúdo dos buffers para representar o estado de um sistema num dado instante de tempo, podendo a mesma ser representada por um vector n-dimensional. Desta representação, surge então a seguinte questão: **será o número de clientes presente em cada um dos buffers a única informação necessária para caracterizar completamente o problema de sequenciamento para servidor único?**

Decorrente de uma análise feita a algumas redes de filas de espera, decidiu-se optar por uma representação do espaço de estados do sistema baseada não só no número de clientes existentes em cada buffer da rede de filas de espera mas também no estado de actividade dos servidores existentes na mesma. A modelação passa então a ser feita de acordo com duas variáveis de $n + k$ posições sendo a variável de n posições referentes ao número de clientes em cada um dos n buffers da rede e a variável de k posições referente ao estado dos k servidores presentes na rede.

De modo a possibilitar o estudo da hipótese levantada, decidiu-se fazer a análise de uma rede de filas de espera simples, com apenas um servidor e duas classes de clientes tal como ilustra a figura seguinte.

Considera-se que o processo de chegada à rede é de Poisson com taxas λ_1 e λ_2 e que os tempos

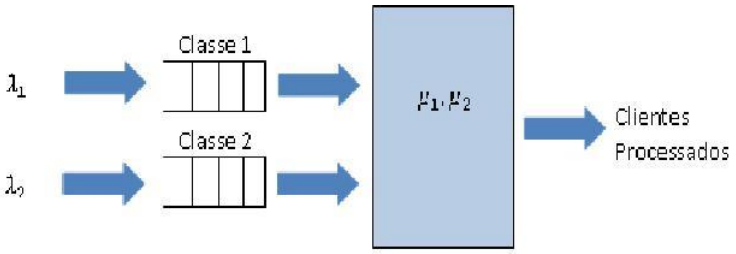


Figura 3.1: Rede de filas de espera com um servidor e duas classes de clientes.

de serviço dos clientes são distribuídos exponencialmente com médias μ_1 e μ_2. Considera-se ainda, tal como referido anteriormente, que os buffers têm capacidade ilimitada e que a rede é aberta, podendo chegar um número ilimitado de clientes à mesma. De acordo com estas considerações e tendo em atenção os conteúdos das secções anteriores, a rede é modelada como uma cadeia de Markov. No que diz respeito à política de sequenciamento, será usada a conhecida μc-rule, referida na secção 2.1.6.

Em geral, para qualquer problema de decisão de Markov, existem transições de estado que implicam a tomada de decisão e existem outras que não conduzem a estados onde seja possível tomar uma decisão. Os pontos de decisão correspondem a instantes de ocorrência de transições que implicam tomada de decisão como sejam os instantes em que ocorra um fim de serviço ou instantes de chegada ao sistema de clientes quando este se encontra vazio. Dado que o serviço é *non-preemptive* uma chegada de um cliente de uma classe prioritária num instante em que esteja a ser processado um cliente de uma classe não prioritária não conduz a um ponto de decisão. Por outro lado quando a rede se encontra vazia, uma vez que o serviço é *non-idling*, o servidor deverá passar a processar o primeiro cliente que chegar à rede não havendo a possibilidade de este permanecer livre.

Estão agora reunidas todas as condições para que se passe à caracterização matemática do problema, ou seja, para que possa ser formulado um problema de decisão de Markov. Irá em seguida proceder-se à formulação do problema primeiramente usando uma variável e posteriormente usando duas variáveis. Finalmente irão ser comparados os dois modelos.

3.1.1 Modelação usando 1 variável

- Caracterização do espaço de estados

$$X(t), X(t) \in \mathbb{N}^2$$

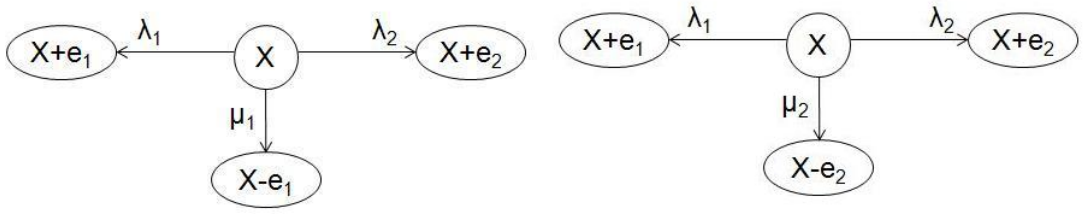
Neste caso o estado é modelado pela variável $X(t)$ que corresponde a um vector de duas posições representando cada uma o número de clientes existentes no buffer 1 e no buffer 2, por esta

mesma ordem;

- Tipos de eventos possíveis:
  - chegada de um cliente ao buffer 1;
  - chegada de um cliente ao buffer 2;
  - conclusão de serviço do tipo 1;
  - conclusão de serviço do tipo 2;

- Esquema de transições em tempo contínuo

  Os esquemas da Figura 3.2 são representativos das cadeias de Markov em tempo contínuo correspondente à rede de filas de espera da Figura 3.1. Uma vez que o serviço é *non-idling*, este deverá estar a processar uma das duas classes de clientes existentes a não ser que não existam cliente na rede. Dependendo da acção de controlo, u, seleccionada, o esquema de transições possível para um instante de tempo poderá corresponder à figura a) ou à figura b). Se por exemplo $u = 1$, o diagrama de transições correspondente será o a). Tal como referido na secção 2.2.4, pode ainda verificar-se que os estados são representados fazendo uso de um vector e não de um inteiro, possibilitando assim o uso das variáveis e_1 e e_2 que correspondem a dois vectores unitários através dos quais é adicionado ou retirado um cliente dos buffers. Ainda a título de exemplo, pode verificar-se que $q_{XX'} = \lambda_1$ quando $X' = X + e_1$.



(a) Diagrama 1. (b) Diagrama 2.

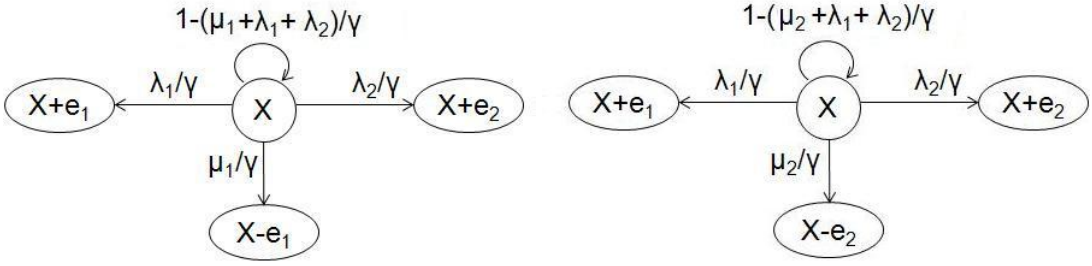
Figura 3.2: Diagramas de transição de estado em tempo contínuo.

- Esquema de transições discreto uniformizado

  Na figura 3.3, γ representa a taxa de transição uniforme e pode ser dada por $\gamma = \mu_1 + \mu_2 + \lambda_1 + \lambda_2$, passando agora as taxas de transição a corresponder a probabilidades de transição. Também neste caso as transições possíveis deverão ser dadas por um dos esquemas da figura 3.3 dependendo da acção de controlo seleccionada num dado instante.

- Critério de Custo

  O critério de custo usado corresponde ao indicado na secção 2.1.6.



(a) Diagrama 1. (b) Diagrama 2.

Figura 3.3: Diagramas de transição de estado uniformizados.

- Algoritmo de PD

Tal como referido, foi usado o algoritmo de Iteração de Valor, obtendo-se a seguinte expressão para cálculo do custo, assumindo um ponto de decisão, onde $x_i > 0$ para $i = 1, 2$:

$$V_{k+1}(X) = \frac{1}{\beta+\gamma}(c_1x_1 + c_2x_2) + \alpha \min\left\{\tilde{V}(X,u|u=0), \tilde{V}(X,u|u=1), \tilde{V}(X,u|u=2)\right\}, \quad (3.1)$$

com,

$$\begin{aligned}
\tilde{V}(X,u|u=0) &= \frac{\lambda_1}{\gamma}V_k(x_1+1,x_2) + \frac{\lambda_2}{\gamma}V_k(x_1,x_2+1) + \left(1-\frac{\lambda_1+\lambda_2}{\gamma}\right)V_k(x_1,x_2),\\
\tilde{V}(X,u|u=1) &= \frac{\lambda_1}{\gamma}V_k(x_1+1,x_2) + \frac{\lambda_2}{\gamma}V_k(x_1,x_2+1) + \frac{\mu_1}{\gamma}V_k(x_1-1,x_2) +\\
&\quad + \left(1-\frac{\lambda_1+\lambda_2+\mu_1}{\gamma}\right)V_k(x_1,x_2),\\
\tilde{V}(X,u|u=2) &= \frac{\lambda_1}{\gamma}V_k(x_1+1,x_2) + \frac{\lambda_2}{\gamma}V_k(x_1,x_2+1) + \frac{\mu_2}{\gamma}V_k(x_1,x_2-1) +\\
&\quad + \left(1-\frac{\lambda_1+\lambda_2+\mu_2}{\gamma}\right)V_k(x_1,x_2).
\end{aligned} \quad (3.2)$$

Apesar de se saber ser a política óptima *non-idling*, optou-se por apresentar como possível a escolha de inactividade na presença de clientes. Na prática, tal opção não nunca surge nas políticas para que se converge.

O vector V é obtido quando o erro associado ao processo iterativo é inferior a um certo valor especificado, ϵ, ou seja,

$$|V_{k+1} - V_k| < \epsilon. \quad (3.3)$$

A expressão de custo 3.1 é válida para o caso em que existem clientes das duas classes na rede. Em alguns pontos de decisão, onde o mesmo não acontece, a expressão 3.1 não assume a forma

exposta, uma vez, que nem todos os termos existem. Por exemplo, no caso em que não existem clientes da classe 2, a expressão 3.1 será dada apenas pelo mínimo dos dois primeiros termos.

3.1.2 Modelação usando 2 variáveis

- Caracterização do espaço de estados

$$X(t), y(t), X(t) \in \mathbb{N}^2, y(t) \in \{0, 1, 2\}$$

Neste caso o estado é modelado pela variável $X(t)$ com o mesmo significado que anteriormente e y(t) que representa o estado do servidor. O valor y(t)=0 corresponde ao caso em que o servidor se encontra livre, y(t)=1 ao caso em que o servidor se encontra a processar um cliente da classe 1 e y(t)=2 ao caso em que o servidor se encontra a processar um cliente da classe 2.

- Tipos de eventos possíveis:

Os eventos são os identificados na secção (3.1.1).

- Esquema de transições em tempo contínuo

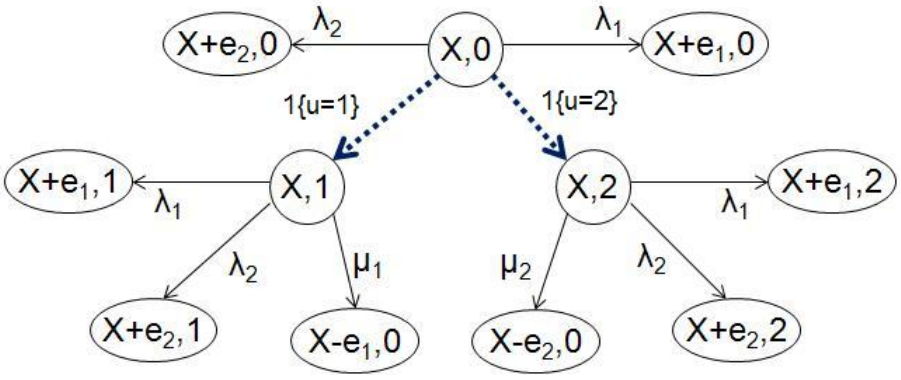


Figura 3.4: Diagrama de Transições do modelo de duas variáveis em tempo contínuo.

O diagrama 3.4 é representativo das cadeia de Markov em tempo contínuo correspondente à rede de filas de espera da Figura 3.1. Neste caso, a não ser que o sistema esteja vazio, o estado deverá corresponder a $\{X, 1\}$ ou a $\{X, 2\}$ dependendo da classe de clientes a ser processada, não correspondendo então as setas a tracejado a uma transição propriamente dita mas sim a transições associadas com a tomada de decisão, por ser necessário representar o novo estado do servidor.

- Esquema de transições discreto uniformizado

Na Figura 3.5 apresenta-se o diagrama de transições, onde γ representa a taxa de transição única e é dada por $\gamma = \mu_1 + \mu_2 + \lambda_1 + \lambda_2$, passando agora as taxas de transição a corresponder

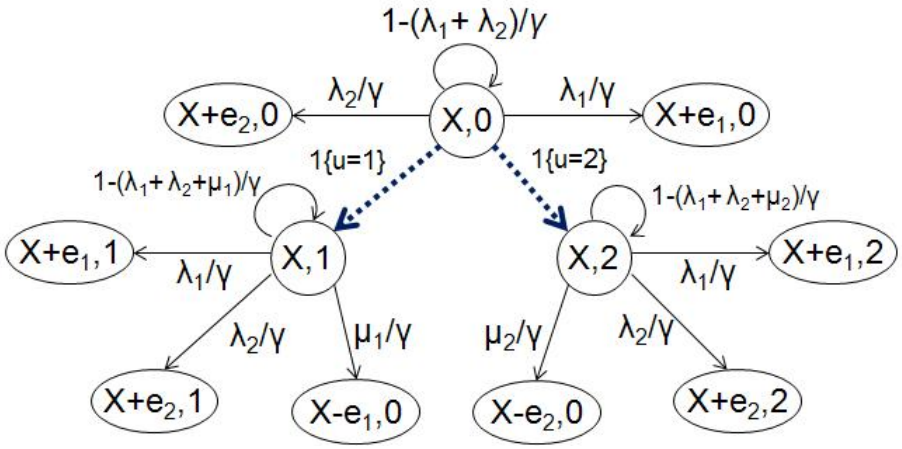


Figura 3.5: Diagrama de Transições do modelo de duas variáveis uniformizado.

a probabilidades de transição. Tal como anteriormente, as variáveis e_1 e e_2 correspondem a dois vectores unitários através dos quais é adicionado ou retirado um cliente dos buffers.

- Critério de Custo

  Tal como no modelo de uma variável, o critério de custo adoptado corresponde ao indicado na secção 2.1.6. Contudo, deve ter-se em conta que neste caso o custo dependerá também do estado do servidor e não apenas do tamanho dos buffers.

- Algoritmo de PD

  Neste caso a expressão que permite a obtenção do custo para um determinado ponto de decisão corresponde a:

$$V_{k+1}(X,0) = \frac{1}{\beta+\gamma}(c_1x_1 + c_2x_2) + \alpha \min\left\{\tilde{V}(X,u|u=0), \tilde{V}(X,u|u=1), \tilde{V}(X,u|u=2)\right\}, \quad (3.4)$$

com,

$$\begin{aligned}
\tilde{V}(X,u|u=0) &= \frac{\lambda_1}{\gamma}V_k(x_1+1,x_2,0) + \frac{\lambda_2}{\gamma}V_k(x_1,x_2+1,0) + \left(1-\frac{\lambda_1+\lambda_2}{\gamma}\right)V_k(x_1,x_2,0),\\
\tilde{V}(X,u|u=1) &= \frac{\lambda_1}{\gamma}V_k(x_1+1,x_2,1) + \frac{\lambda_2}{\gamma}V_k(x_1,x_2+1,1) + \frac{\mu_1}{\gamma}V_k(x_1-1,x_2,0) +\\
&\quad + \left(1-\frac{\lambda_1+\lambda_2+\mu_1}{\gamma}\right)V_k(x_1,x_2,1), \qquad (3.5)\\
\tilde{V}(X,u|u=2) &= \frac{\lambda_1}{\gamma}V_k(x_1+1,x_2,2) + \frac{\lambda_2}{\gamma}V_k(x_1,x_2+1,2) + \frac{\mu_2}{\gamma}V_k(x_1,x_2-1,0) +\\
&\quad + \left(1-\frac{\lambda_1+\lambda_2+\mu_2}{\gamma}\right)V_k(x_1,x_2,2).
\end{aligned}$$

Tal como no modelo de uma variável, dependendo dos valores de x_1 e x_2, a expressão do custo poderá não ser dada pelo mínimo dos três termos.

As expressões 3.6 e 3.7, correspondem ao custo para estados em que a decisão de processar clientes de uma dada classe já foi tomada mas que têm que ser contabilizados na aplicação do algoritmo.

$$\begin{aligned} V_{k+1}(x_1,x_2,1) &= \frac{1}{\beta+\gamma}(c_1x_1+c_2x_2)+\ \alpha\left\{\frac{\lambda_1}{\gamma}V_k(x_1+1,x_2,1)+\frac{\lambda_2}{\gamma}V_k(x_1,x_2+1,1)+\right. \\ &\left.+\ \frac{\mu_1}{\gamma}V_k(x_1-1,x_2,1)+\left(1-\frac{\lambda_1+\lambda_2+\mu_1}{\gamma}\right)V_k(x_1,x_2,0)\right\}, \end{aligned} \tag{3.6}$$

$$\begin{aligned} V_{k+1}(x_1,x_2,2) &= \frac{1}{\beta+\gamma}(c_1x_1+c_2x_2)+\ \alpha\left\{\frac{\lambda_1}{\gamma}V_k(x_1+1,x_2,2)+\frac{\lambda_2}{\gamma}V_k(x_1,x_2+1,2)+\right. \\ &\left.+\ \frac{\mu_2}{\gamma}V_k(x_1-1,x_2,2)+\left(1-\frac{\lambda_1+\lambda_2+\mu_2}{\gamma}\right)V_k(x_1,x_2,0)\right\}. \end{aligned} \tag{3.7}$$

Também neste caso o vector V é obtido quando o erro associado ao processo iterativo é inferior a um certo valor especificado, ϵ, ou seja,

$$|V_{k+1}-V_k|<\epsilon. \tag{3.8}$$

A expressão 3.6 permite calcular o custo para estados em que se optou por processar clientes da classe 1 e a expressão 3.7 ao caso em que se optou por calcular o custo para estados em que se decidiu processar clientes da classe 2. A expressão 3.6 só é válida para $x_1>0$ e a expressão 3.7 para $x_2>0$. Nestes casos, como é lógico, os estados não correspondem a pontos de decisão.

3.1.3 Comparação dos Modelos

Como pode verificar-se pela comparação dos dois modelos, diversas diferenças podem ser tidas em conta. Observando os diagramas de transição, pode verificar-se que, embora os eventos possíveis de acontecer num modelo e noutro sejam os mesmos, existem estados que sendo diferentes são representados no modelo de uma variável, da mesma forma. No que diz respeito ao custo, no caso do modelo de duas variáveis este passa a depender não só dos tamanhos dos buffers da rede, mas também do estado do servidor, como pode ser facilmente verificado pela comparação das expressões de custo.

Considere-se a figura 3.6 na qual podem ser visualizados os eventos possíveis de acontecer num dado estado do sistema.

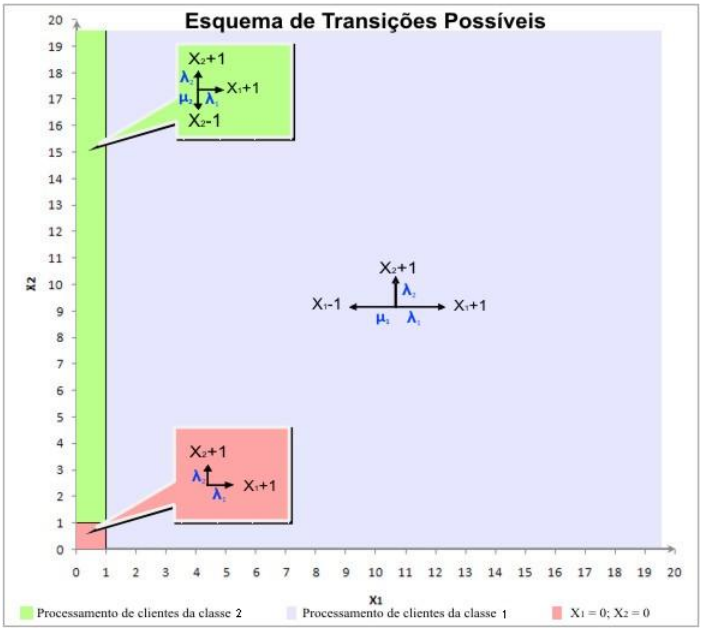


Figura 3.6: Matriz de decisão da μc-rule e transições possíveis.

A figura 3.6 mostra a matriz de decisão da μc-rule onde a classe prioritária é a classe 1. Como pode ser verificado, através das setas que reflectem as transições possíveis, na zona azul, onde existem clientes de ambas as classes, a prioridade de serviço é dada à classe 1 não podendo ocorrer um evento do tipo conclusão de serviço da classe 2.

Como é de fácil constatação, as transições não controláveis, ou seja, as chegadas, podem fazer com que o sistema transite de uma zona onde é óptimo processar clientes de uma classe para uma outra zona onde é óptimo processar clientes de uma outra classe. É neste ponto que é possível estabelecerem-se diferenças entre o modelo usando apenas uma variável com o modelo usando duas variáveis. Se considerarmos o modelo de apenas uma variável, não é possível saber-se qual é a classe que está a ser processada, a não ser em alguns estados muito particulares, mesmo que se saiba em que zona do gráfico se está num dado momento. Contrariamente, o mesmo não acontece no modelo de duas variáveis.

Como a política é do tipo *non-idling* o servidor encontra-se sempre em processamento a não ser que não existam clientes. Por exemplo, se o estado indicado for $X(t) = [2, 2]$, não nos é possível no modelo de uma variável saber se o processador se encontra a processar um cliente da classe 2 ou da classe 1 uma vez que é possível atingir-se este estado tanto após conclusão de um serviço da classe 1 como da classe 2. O estado pode ser atingido por conclusão de um serviço da classe 2, partindo-se do estado $X(t) = [2, 3]$, ou então partindo do estado $X(s|s < t) = [0, 3]$, onde a decisão óptima corresponde a processar clientes da classe 2, podendo, neste caso chegar dois clientes da classe 1 enquanto se processa um cliente da classe 2, atingindo-se seguidamente o estado $X(t) = [2, 2]$. No primeiro caso, a transição controlável correspondente ao fim de serviço de um cliente de classe 1 acontece com taxa

μ_1, no segundo caso, a transição controlável corresponde ao fim de serviço de um cliente de classe 2, com taxa μ_2. Como visto anteriormente, não é então possível saber-se no modelo de uma variável qual das transições irá acontecer. Pode ser verificado a partir da equação 3.1 que o custo não é calculado de acordo com a acção que está a ser executada, o que está em concordância com o anteriormente dito.

Pelo contrário, no caso da modelação usando duas variáveis esta dúvida não surgiria, ou se estaria no estado $\{X(t), y(t)\} = \{2, 2; 1\}$ ou no estado $\{X(t), y(t)\} = \{2, 2; 2\}$, sendo neste caso possível saber-se qual a acção que está a ser executada num dado instante de tempo. Em forma de conclusão, pode então verificar-se que no caso da modelação usando apenas uma variável, os dois estados anteriores são equivalentes, o que como é de fácil compreensão, não corresponde exactamente à realidade.

3.2 A $\mu\triangle$-Rule

Embora este tema tenha sido o principal impulsionador desta tese, foram usados como ferramenta no desenvolvimento deste tema os resultados obtidos do tema anteriormente exposto. Ou seja, todos os modelos usados no estudo deste problema foram representados usando duas variáveis (tal só foi possível uma vez que as conclusões do tema anterior já tinham sido obtidas).

Conhecido o comportamento das políticas de sequenciamento que fazem uso de prioridades estritas, colocou-se a hipótese de formular um problema de sequenciamento em filas de servidor único que fizesse uso de um critério de custo que possibilitasse uma maior equidade na distribuição das prioridades das classes existentes. O critério de custo adoptado é semelhante ao usado na μc-rule mas com a diferença de contemplar, não só uma componente linear, mas também uma componente quadrática. A introdução de uma componente quadrática que actue em sentido contrário ao parâmetro linear deve então produzir um efeito que tem como consequência uma maior equidade no serviço de atendimento das diferente classes pelo servidor.

Assim, tal como no problema de modelação, decidiu-se fazer o estudo de uma rede de pequenas dimensões tendo-se optado por utilizar a mesma rede usada no problema anterior, representada na figura 3.1, bem como os mesmos modelos para as chegadas à rede e para os tempos de serviço. Quanto à formulação do problema, a caracterização do problema de decisão de Markov é feita de modo semelhante ao feito na secção 3.1.2, sendo usada a modelação com duas variáveis como referido. O critério de custo é também o mesmo, ou seja, baseado no custo total descontado em horizonte infinito. A única diferença a ser considerada passa então pela equação que traduz o custo indexado a cada estado:

$$C\left[X\left(t\right),y\left(t\right)\right]=a_1x_1(t)+b_1\left[x_1(t)\right]^2+a_2x_2(t)+b_2\left[x_1(t)\right]^2 \tag{3.9}$$

onde $a_1, b_1, a_2\ e\ b_2$ são as taxas do custo. A expressão de custo passa agora a depender também de um factor quadrático, contrariamente ao que acontecia com a μc-rule.

Tal como no problema da modelação, é usado o algoritmo de Iteração de Valor onde, de acordo com o referido anteriormente, deverá figurar a nova expressão de custo.

O objectivo passa então por determinar qual a nova acção a ser tomada que conduz a um custo mais baixo que por sua vez leva à determinação de uma nova política. Da formulação do problema de minimização de custo deverá saber-se qual a acção que conduz ao menor custo, podendo esta corresponder a processar clientes da classe 1, da classe 2 ou então a não fazer nada. Desta minimização deverá então resultar uma matriz de decisão que deverá indicar qual a acção mais favorável a ser realizada um dado estado do espaço de estados.

Pode ainda dizer-se que o principal objectivo passa por verificar se a **introdução deste novo critério de custo decorrente da variação dos parâmetros a_1, b_1, a_2 e b_2 realmente produz uma maior equidade na distribuição das prioridades das classes.** Este facto será verificado calculando a variância do tempo de espera de cada uma das classes.

Capítulo 4

Simulação

Passadas as etapas de formulação teórica do problema, o objectivo seguinte é obter quantidades de interesse relativamente ao modelo em causa. No presente caso, as quantidades de interesse correspondem aos tempos de entrada e de saída dos clientes nos buffers e da rede, respectivamente, bem como o número de clientes existentes em cada um dos buffers e no servidor em cada um dos instantes anteriores. A simulação é então tida como um processo pelo qual um modelo de um sistema é avaliado numericamente através da estimação de diferentes quantidades de interesse, para posterior cálculo das medidas de desempenho.

A simulação computacional pode ser assemelhada a um bom laboratório experimental onde o único hardware realmente é um computador. Embora a simulação não seja exactamente a realidade, é a melhor aproximação que se tem uma vez que a maior parte dos sistemas são impossíveis de ser testados.

Nas secções seguintes deste mesmo capítulo serão apresentadas as características gerais do pacote de simulação implementado bem como a sua arquitectura e funcionalidades.

Este simulador foi desenvolvido mais significativamente por [18] tendo havido contribuições importantes na concepção de desenvolvimento como parte do trabalho que se descreve nesta dissertação. Para mais detalhes relativos ao simulador, consultar [18].

4.1 Características Gerais

Tendo em vista a possível continuidade do estudo dos problemas desenvolvidos nesta tese ou mesmo de novos problemas, optou-se por conceber um pacote de simulação que possibilitasse não só a simulação da rede em estudo nesta tese mas de qualquer rede de actividades. Para que possa ser mais

facilmente entendido o grau de generalização deste pacote, irá proceder-se a uma breve descrição das características das redes de actividades.

As redes de actividades, tal como as redes de filas de espera são modeladas usando sistemas de eventos discretos. As redes de filas de espera são um caso particular das redes de actividades, podendo todas as características referidas no capítulo 2 aplica-se às redes de actividades. No caso das redes de actividades, tal como referido no início deste texto, na Introdução, os clientes são chamados de materiais e os servidores de máquinas. As redes de actividades possuem como principais características:

- split: a máquina tem a possibilidade de a partir de uma única unidade de material produzir, outras duas ou mais, unidades de um outro tipo de material;
- merge: a máquina tem a possibilidade de reunir uma ou mais unidades de material, de um mesmo buffer ou de diversos buffers, e produzir destas mesmas unidades apenas uma unidade de outro material;
- trabalho em equipa: pode existir a possibilidade de várias máquinas trabalharem em conjunto numa mesma actividade para obter um mesmo fim.

Contudo, nem todas as propriedades acima enunciadas foram programadas, não podendo portanto ser utilizada a opção de trabalho em equipa.

O pacote de simulação (simulador) foi desenvolvido fazendo uso de uma linguagem orientada a objectos, o JAVA [1]. Graças às propriedades deste tipo de programação, é possível ao simulador a criação de objectos virtuais, com características especificadas pelo utilizador, que por sua vez proporcionam a criação, também virtual, da rede de filas de espera que se quer simular. As especificações do modelo deverão fazer o mapeamento com os objectos virtuais criados e os seus respectivos atributos. Nas secções seguintes, este mapeamento será explicado mais pormenorizadamente. Também as políticas a testar poderão ser especificadas pelo utilizador do programa.

4.2 Modelação da Rede

Nesta secção irá proceder-se à descrição dos modelos que possibilitaram a construção do simulador. Estes modelos foram construídos com base nos modelos desenvolvidos por Harrison [13, 14], e tal como no caso dos modelos teóricos atrás referidos, três tipos de objectos são considerados: clientes, servidores e buffers. A dinâmica entre estes mesmos elementos é realizada de acordo com actividades que por sua vez obedecem a políticas de sequenciamento.

Assume-se que existem l servidores, j actividades possíveis de serem realizadas e i buffers (de entrada que recebem materiais vindos do exterior do sistema com taxas médias de chegada λ_i e de serviço ou intermédios que recebem materiais acabados de ser processados nos servidores). Cada actividade j usa um servidor particular com uma taxa média de processamento μ_j de uma unidade de material por unidade de tempo. Como visto anteriormente, estes materiais podem ser retirados de um ou mais buffers e posteriormente posicionados nos buffers de saída correspondentes a essa mesma actividade.

Considere-se a matriz R_{ij}, através da qual é possível saber qual a quantidade de material consumida/inserida do/no buffer i pela actividade j. As posições da matriz podem conter valores inteiros negativos e positivos, assumindo-se que quando o valor é negativo o buffer recebe material resultante da conclusão de uma actividade e que quando o valor é positivo são retiradas as unidades de material correspondentes a esse valor para processamento. Considera-se ainda uma segunda matriz A_{kj}, onde cada posição representa a taxa à qual a actividade j consome capacidade do recurso k.

Finalmente, consideram-se ainda quatro vectores: L, Q , $costs$ e $politicas$. O vector L deve ter dimensão igual ao número de buffers existentes na rede sendo contudo apenas preenchidas as posições respeitantes a buffers de entrada do sistema, uma vez que este vector contém as médias das taxas de chegada de cada uma das classes de materiais existentes à rede. O vector Q deve ter dimensão igual ao número de servidores existentes na rede, sendo os valores destas mesmas posições preenchidos apenas por valores inteiros não negativos, dado este vector possibilitar a criação de cópias de cada um dos servidores existentes. O vector $costs$ deve ter dimensão igual ao número de classes de materiais existentes na rede, uma vez que é a partir deste vector que a política de sequenciamento $\mu c - rule$ pode ser simulada. Por fim, o vector $politicas$ corresponde a um vector de apenas uma posição, sendo este preenchido com o número correspondente à política que se quer simular, uma vez que todas as políticas implementadas têm um valor que as indexa.

No caso da Figura 4.1, a criação da rede pode ser feita de acordo com os seguintes parâmetros:

- $l = 1$
- $j = 2$
- $i = 4$
- $L = [0.8\ 1.1]$
- $A = [0.5\ 1.5]$

- $R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}$
- $Q = [1]$
- $costs = [2\ 1]$
- $politicas = [5]$

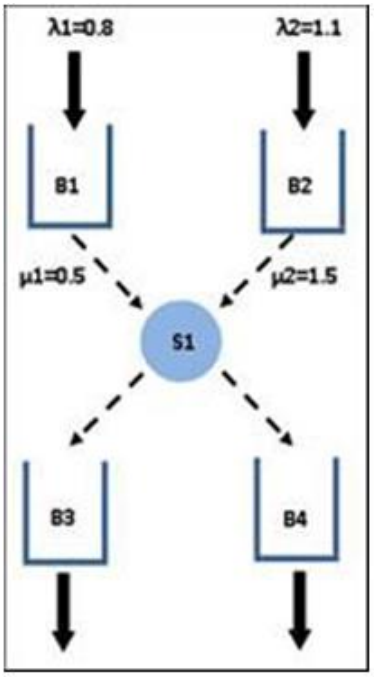


Figura 4.1: Rede de filas de espera.

Como pode verificar-se, a rede da Figura 4.1 pode realizar duas actividades, com taxas de processamento inicadas pelo vector A e inserir/retirar os materiais resultantes dessas mesmas actividades de quatro buffers de acordo com a matriz R. Por exemplo, a actividade 1 é processada a uma taxa $\mu_1 = 0.5$, retira uma unidade de material por cada processamento do buffer 1, ($R(0,0)$), e coloca a unidade de material resultante de um serviço no buffer 3, ($R(2,0)$). O vector Q indica que existe apenas um máquina, não havendo máquinas idênticas. A política a ser executada corresponde à política indexada pelo número 5 e pode usar, se for caso disso, o vector de custos, $costs$.

4.3 Dinâmica da Rede

A dinâmica da rede é executada através da criação e posterior processamento de eventos. O simulador é constituído por uma estrutura chamada Cadeia de Acontecimentos Pendentes (CAP) onde os eventos

são posicionados à medida que vão sendo criados e retirados após serem processados. Quando os eventos são criados, independentemente do seu tipo, é lhes preenchido um atributo que indica o seu tempo de execução. É a partir deste atributo que os eventos são organizados na CAP e posteriormente retirados por essa mesma ordem, sendo a partir da execução destes eventos que o tempo da simulação evolui. Os eventos podem ser de dois tipos:

- Eventos de chegada: correspondem à chegada de material do exterior da rede. Quando um evento destes é executado, uma unidade de material é criada e posicionada num buffer. É também verificado se alguma actividade pode ser iniciada. Em caso afirmativo um evento fim de serviço é gerado e posicionado na CAP.

- Evento fim de serviço: quando um evento deste tipo é executado, significa que uma máquina acabou de processar material sendo o material resultante deste processamento posicionado num buffer de acordo com a actividade que acabou de ser efectuada. Seguidamente é verificado se é possível a este mesmo servidor, que acabou de ficar livre, realizar uma outra actividade. Também se verifica se alguma outra actividade pode ter inicio por ter chegado ao buffer de saída deste que terminou.

4.4 Especificação das Políticas

No contexto deste trabalho, tal como na programação dinâmica, o objectivo da simulação é explorar a performance de políticas de sequenciamento. As políticas podem ser seleccionadas de uma lista de políticas previamente programadas no simulador. Esta escolha é feita tal como foi referido anteriormente através da especificação do número da política que se quer simular pelo preenchimento do vector políticas com o número correspondente à mesma. Novas políticas podem ser acrescentadas ao código uma vez que cada política corresponde a um método da classe simulador, embora esta actualização implique que as mesmas sejam programadas de raiz.

4.5 Arquitectura do Simulador

Todos o parâmetros anteriormente mencionados $(L, Q, R, A, ...)$ são definidos fazendo uso de um ficheiro de entrada em formato txt. A partir deste ficheiro de entrada e de acordo com a programação orientada a objectos, os objectos virtuais são criados com base nas *templates* existentes e os seus atributos são preenchidos. O UML (*Unified Modelling Language*) parcial da Figura 4.2 ilustra a arquitectura do simulador bem como as relações entre os objectos, podendo ser consultado o UML total do

simulador em anexo. O simulador é o objecto principal do programa, uma vez que é a partir deste que

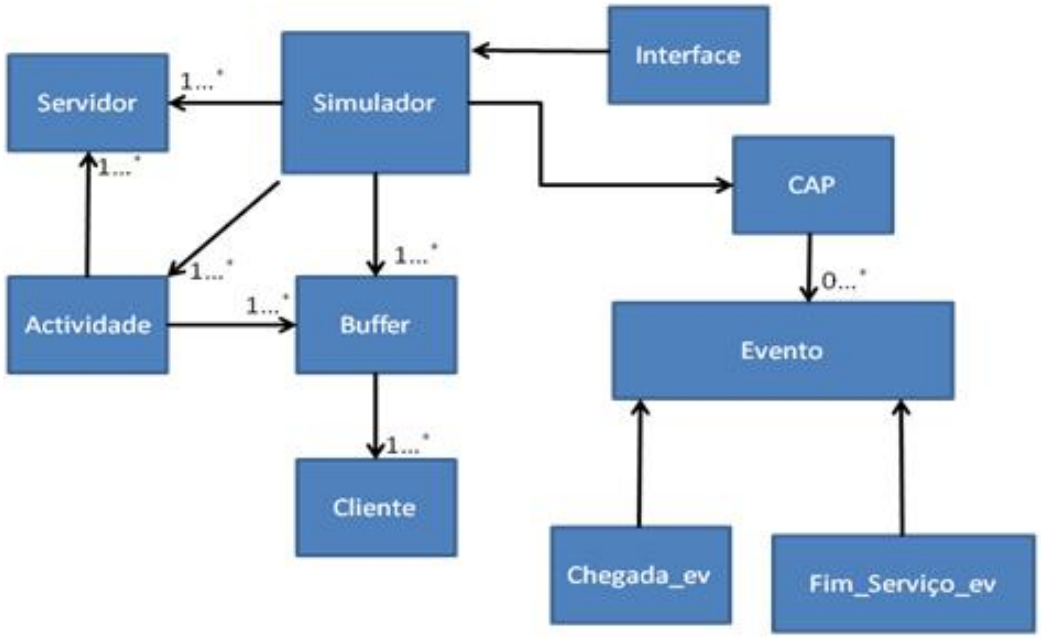


Figura 4.2: UML.

a rede virtual é criada. A simulação é iniciada e finalizada, a execução dos eventos é levada a cabo e as políticas são definidas. Embora todas estas tarefas sejam efectivamente realizadas ao nível do objecto simulador, o objecto interface possibilita a escolha do ficheiro de entrada, o critério de paragem da simulação bem como o número de simulações que se pretendem realizar. A interface apresentada ao utilizador tem então o aspecto da Figura 4.3. O critério de paragem poderá ser definido de três formas

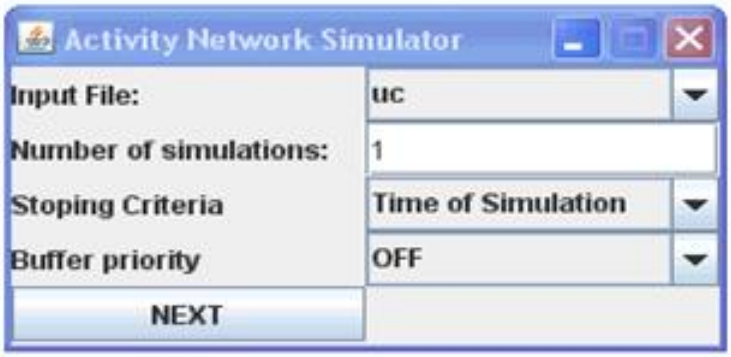


Figura 4.3: Interface.

distintas: por número de clientes processados, por tempo de simulação ou por número de intervalos de regeneração. Define-se que se tem um intervalo de regeneração sempre que a rede fica vazia, ou seja, quando não existe nenhum cliente na rede.

4.6 Recolha de Informação

De modo a não sobrecarregar o simulador com cálculos, decidiu-se que toda a informação obtida pela simulação seria processada numa segunda fase e do modo mais conveniente para o utilizador. Optou-se por guardar em ficheiros txt a informação relevante para o estudo em causa, correspondendo esta a atributos dos diferentes objectos. Neste caso particular, a informação relevante corresponde aos tempos de entrada e saída dos clientes da rede bem como o número de clientes em cada um dos buffers nos instantes anteriormente citados. A Figura 4.4 mostra um exemplo de um ficheiro de saída.

ID do Buffer	Hora de chegada	Hora de Saída	Nº Clientes no buffer 1	Nº Clientes no buffer 2
2	1,462470		0	1
1	2,119891	0	1	1
2	0	2,690549	1	0
2	2,927961	0	1	1
2	3,647468	0	1	2
1	0	3,892568	0	2
1	3,916174	0	1	2
1	5,219984	0	2	2
2	5,772085	0	2	3
2	6,105219	0	2	4
1	7,502618	0	3	4
2	8,383452	0	3	5

Figura 4.4: Tabela de output do simulador.

A informação recolhida, dependo do utilizador, poderá ser trabalhada em Matlab, Excel ou outra ferramenta com a qual o utilizador se sinta mais familiarizado ou que corresponda melhor ao desejado. No caso deste trabalho decidiu-se por usar o Matlab. Resultados exemplificativos do funcionamento do simulador podem ser consultados em [19].

Capítulo 5

Resultados

Neste capítulo, serão evidenciados os resultados obtidos por simulação e por programação dinâmica das duas hipóteses em estudo. Na primeira secção, fazer-se-á referência aos resultados obtidos pela modelação de redes de filas de espera usando uma e duas variáveis e, posteriormente, será feita uma comparação entre os valores obtidos no caso em que foi usada programação dinâmica como o caso em que foi usada a simulação. Na segunda secção serão apresentados os resultados relativos à obtenção da nova política de sequenciamento, mais uma vez usando os valores obtidos através de programação dinâmica e da simulação. Ainda respeitando a este tema, foi feita uma abordagem intuitiva para obtenção da política e foram comparados os resultados dessa mesma abordagem com os anteriormente conseguidos. Finalmente, são apresentados resultados que expressam a melhoria da variância global do tempo de ciclo das classes de clientes do sistema com a nova política em relação à μc-rule.

5.1 Modelação usando 1 Variável VS 2 Variáveis

Tal como foi referido na secção 3.1, é pretendido com a formulação deste problema provar, por métodos numéricos, que a modelação de alguma redes de filas de espera usando apenas uma variável não corresponde a uma representação fiel. O objectivo é então obter quantidades de interesse para os dois tipos de modelação fazendo uso de PD e seguidamente fazer a simulação da rede e obter a mesma quantidade de interesse. Por fim, a conclusão acerca de qual das teorias poderá estar correcta advém da comparação dos valores obtidos por PD com os valores obtidos por simulação. A quantidade de interesse a calcular será então o custo de alguns estados do sistema de acordo com um critério baseado no custo total descontado esperado num horizonte infinito.

Embora o critério seja baseado em horizonte infinito tanto a simulação como o cálculo dos custos por PD foram feitos num número finito de iterações de acordo com o critério atrás referido.

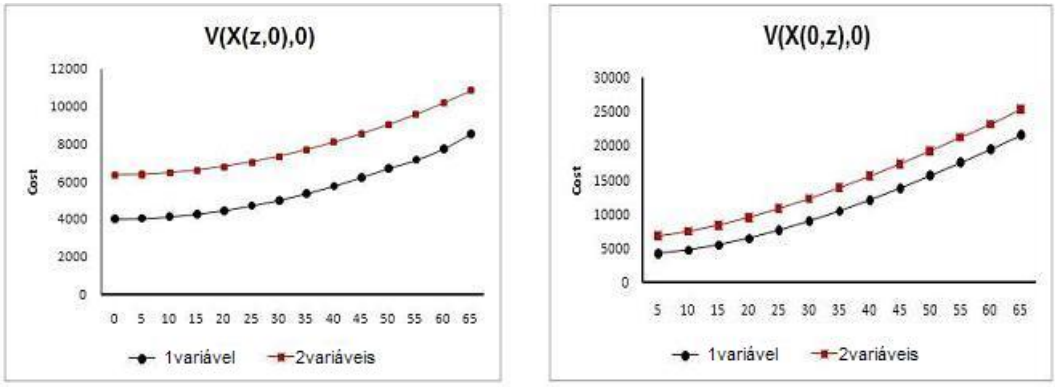
5.1.1 Programação dinâmica

De acordo com o formulado na secção 3.2, obtiveram-se custo de diferentes estados do espaço de estados, usando a política μc-rule com os seguintes parâmetros:

$$\begin{cases} \lambda_1 = 3, \mu_1 = 4, c_1 = 2, \\ \lambda_2 = 0.666666, \mu_2 = 0.1, c_2 = 1 \\ \beta = 0.002. \end{cases}$$

Pode verificar-se usando a Definição 2 da secção 2.1.5 que os parâmetros acima indicados permitem a estabilidade da rede.

Calculados os custos, obtiveram-se os gráficos 5.1.



(a) Variação do custo com x_1.

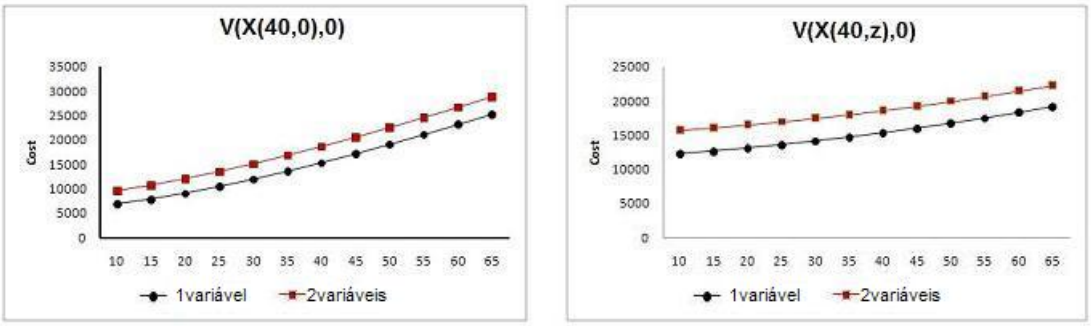
(b) Variação do custo com x_2.

Figura 5.1: Custo de estados iniciais do espaço de estados.

Os gráficos da Figura 5.1 representam o custo obtido para diferentes estados quando se mantém constante o número de cliente no buffer um ou dois, dependendo da figura da direita ou da esquerda e se vai aumentando o número de clientes do outro buffer. Por exemplo, no gráfico da direita da Figura 5.1, aumenta-se x_1 mantendo-se x_2 igual a zero.

Como pode verificar-se pela análise dos gráficos, o custo obtido fazendo a modelação usando apenas uma variável é substancialmente inferior ao obtido no caso de ser usado um modelo com duas variáveis. Verifica-se ainda que o comportamento do custo tem uma evolução monotónica correspondendo estados com maior número de clientes a um maior custo como é de fácil compreensão.

O comportamento dos gráficos da Figura 5.2 é idêntico ao obtido nos gráficos da Figura 5.1, corres-



(a) Variação do custo com x_1.

(b) Variação do custo com x_2.

Figura 5.2: Custo de estados centrais do espaço de estados.

pondendo mais uma vez ao resultado esperado. Desta vez os gráficos são referentes a uma área mais central do espaço de estados, pretendendo-se demonstrar que as características dos resultados não se verificam apenas numa secção do espaço de estados mas sim em todo ele.

5.1.2 Simulação da μc-rule

Nesta secção, é apresentada toda a informação relativa à simulação da μc-rule para a rede especificada na secção 3.1. O ficheiro de entrada que permite a simulação da rede especificada dando prioridade à classe 1 de acordo com os parâmetros $c_1 = 2$, $c_2 = 1$ e $\mu_1 = 4$ e $\mu_2 = 0.666666666$ para a classe 1 e para a classe 2 respectivamente, tem a seguinte forma:

```
servidores 1;
buffer 2;
actividade 2;
A=[4 0.666666666];
R=[1 0 0 1];
L=[3 0.1];
Q=[1];
custos=[2 1];
politicas=[4];
```

Figura 5.3: Ficheiro de entrada.

O ficheiro de entrada corresponde à criação da rede ilustrada na Figura 5.4:

As únicas especificações que não são evidenciadas na Figura 5.4 dizem respeito à especificação da política, a qual já foi referida, e ao número de unidades de material consumidas por cada actividade, que neste caso corresponde a uma unidade, como pode ser visto através da matriz R.

No que diz respeito ao critério de paragem, foi usado em todas as simulações o número de intervalos de regeneração, sendo parada a simulação quando são atingidos 1000 intervalos de regeneração.

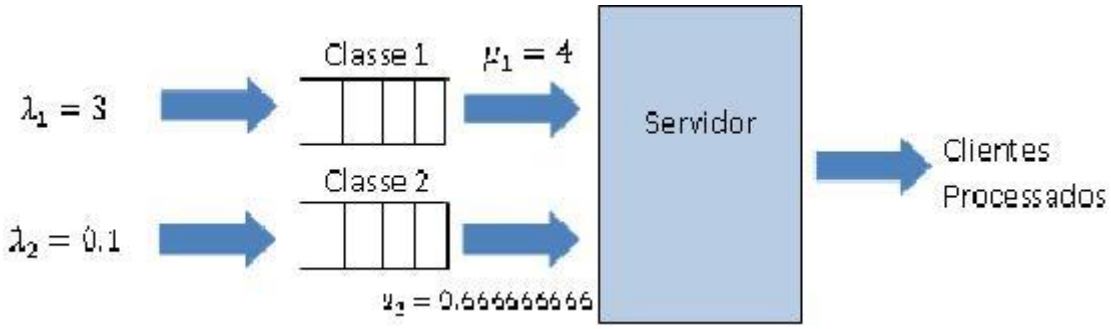
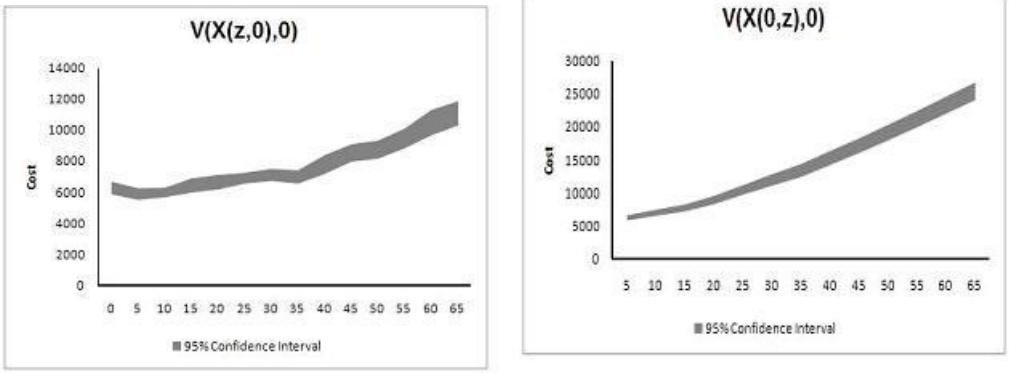


Figura 5.4: Rede de filas de espera virtual criada na simulação.

No caso do número de simulações optou-se por efectuar 25 simulações, cada uma com uma *seed* diferente. Quando se pretende simular a rede com diferentes parâmetros, as mesmas 25 *seeds* são usadas, correspondendo a cada uma destas *seeds* uma simulação. Ambos os critérios foram passados ao simulador através da interface, a qual é ilustrada na figura 4.3.

O processamento da informação obtida corresponde ao cálculo dos custos de estados do sistema com base no ficheiro de saída do simulador. O custo de cada estado é calculado através do integral descontado do número de clientes ao longo do tempo de simulação, de acordo com a expressão 2.23. No que diz respeito ao factor de desconto, é usado $\beta = 0.002$ tal como no caso da PD.

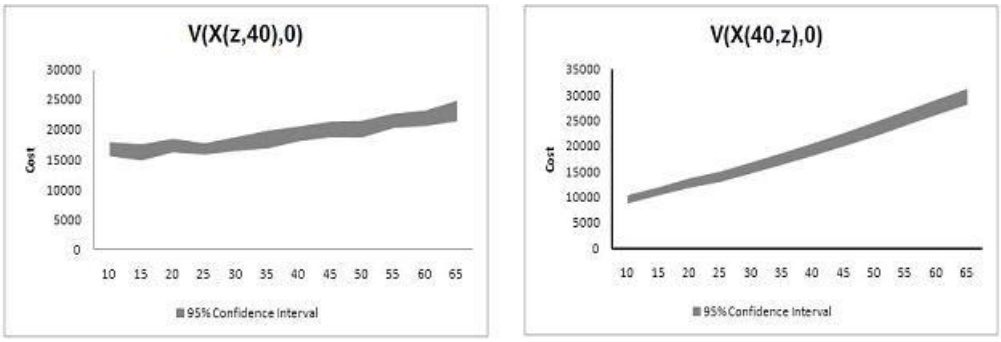
Finalmente, com os valores obtidos é então construído um intervalo de confiança a 95%, obtendo-se os resultados das figuras 5.5 e 5.6.



(a) Variação do intervalo de confiança com x_1.

(b) Variação do intervalo de confiança com x_2.

Figura 5.5: Intervalo de confiança a 95% para estados inciais do espaço de estados.

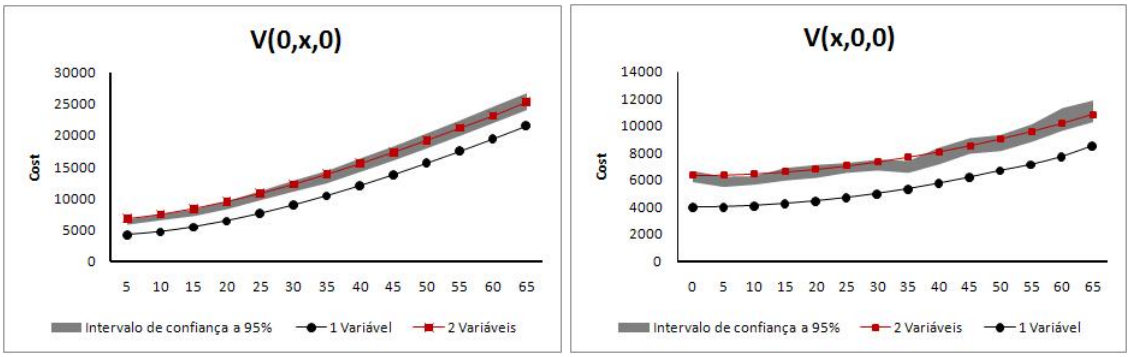


(a) Variação do intervalo de confiança com x_1. (b) Variação do do intervalo de confiança com x_2.

Figura 5.6: Intervalo de confiança a 95% para estados inciais do espaço de estados.

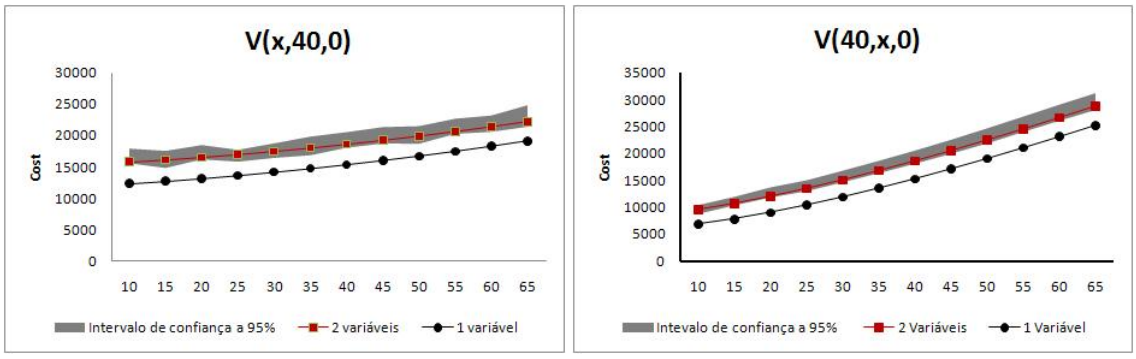
5.1.3 Comparação de Resultados

Como modo de comparação dos resultados resta agora fazer a sobreposição dos resultados obtidos com a PD e a simulador.



(a) (b)

Figura 5.7: Sobreposição dos custos obtidos por PD no intervalo de confiança.



(a) (b)

Figura 5.8: Sobreposição dos custos obtidos por PD no intervalo de confiança.

Destes resultados, pode facilmente retirar-se a conclusão que o modelo de duas variáveis é o mais adequado para modelação de redes de filas de espera deste género. Independentemente do estado

em que o sistema se encontre, os valores obtidos por PD quando é feita a modelação usando duas variáveis encaixam quase todos no intervalo de confiança obtido através da simulação, validando a conclusão acima. As raras excepções em que tal não se verifica não têm a ver com uma falha no cálculo dos valores da PD mas sim com os resultados obtidos por simulação. Um modo de diminuir este erro passaria por realizar um maior número de simulações do sistema, ou seja, em vez de fazer 25 simulações como foi realizado no caso deste trabalho, fazer por exemplo 100 ou mais simulações e/ou por aumentar o número de intervalos de confiança.

5.2 A $\mu\Delta - Rule$

De acordo com o anteriormente exposto, o objectivo é obter uma política de sequenciamento que faça uma distribuição mais equitativa das prioridades das classes, do que as políticas que fazem uso de prioridades estritas.

5.2.1 Resultados da PD

Resolvendo o problema de minimização de custo para cada estado do sistema, recorrendo à PD e posteriormente obtendo a matriz de decisão, é possível, caso os resultados da hipótese formulada sejam coerentes e de possível formulação teórica obter uma nova política de sequenciamento.

As matrizes de decisão são obtidas escolhendo iterativamente a acção que minimiza o custo de cada estado, ou seja, caso o custo mais baixo seja obtido se for escolhido processar clientes da classe 1 num determinado estado, então, figurará na correspondente posição da matriz de decisão o número correspondente a essa acção: 0 no caso de a inactividade ser mais favorável, 1 no caso de ser favorável processar clientes da classe 1 e 2 no caso de ser mais favorável processar clientes da classe 2. Optou-se por obter matrizes de decisão para o modelo com custos lineares e quadráticos com diferentes valores das componentes quadráticas mas mantendo sempre os valores dos custos lineares iguais a zero. Consideraram-se então os seguintes parâmetros:

$$\begin{cases} \lambda_1 = 0.475, \mu_1 = 1, a_1 = 0, b_1 = 0.5 \\ \lambda_2 = 0.475, \mu_2 = 1, a_2 = 0, b_2 = ? \\ \beta = 0.002 \end{cases}$$

Nas figuras que se seguem, podem ver-se matrizes de decisão obtidas com diferentes valores de b_2.

Como pode ser verificado pela comparação das matrizes de decisão da Figura 5.9, com o aumento

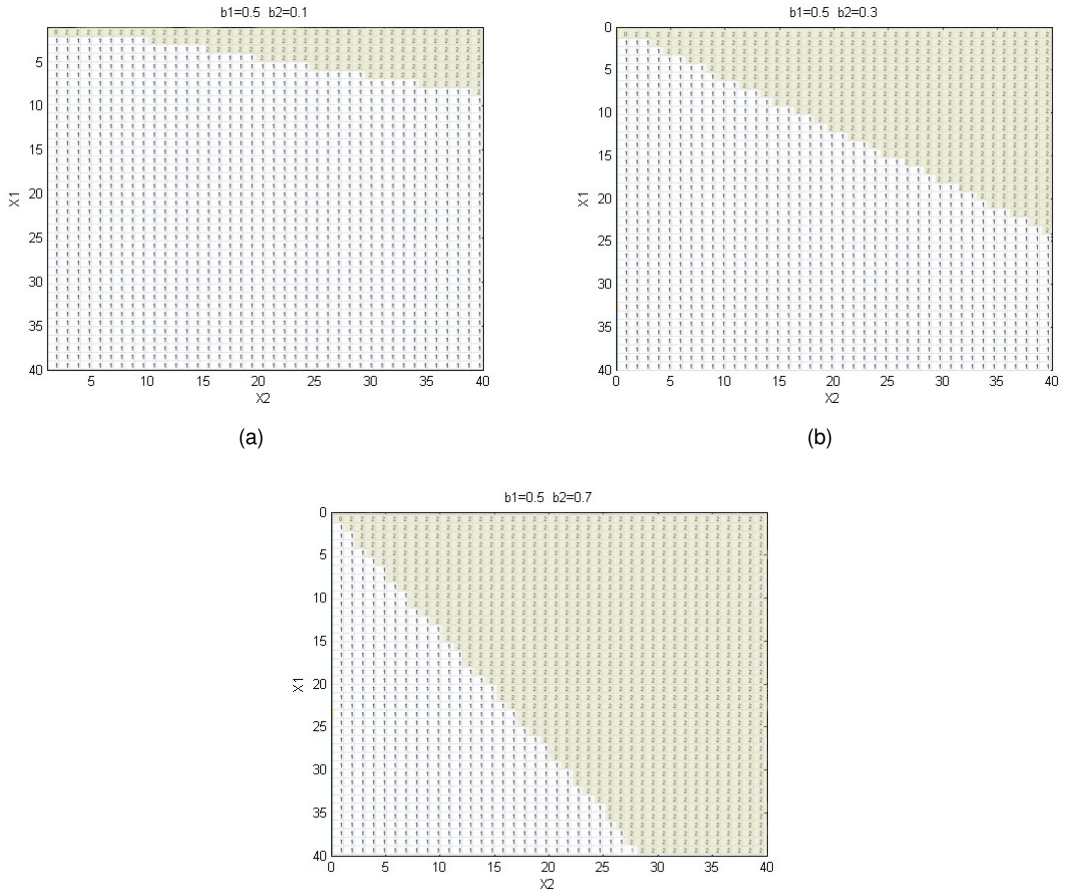


Figura 5.9: Matrizes de decisão.

de b_2, a prioridade da classe 2 aumenta enquanto que a prioridade da classe 1, como seria de esperar, diminui. Para além disso, embora não sejam apresentados muitos resultados, pode ver-se de uma matriz para outra que a política parece ter um comportamento de decisão linear, intuindo-se portanto a possibilidade de formulação teórica da política.

5.2.2 Formulação Intuitiva da Política

A primeira abordagem realizada foi no sentido de se encontrar uma política que correspondesse à derivada da expressão do custo. No entanto, veio a verificar-se que essa hipótese coincidindo com a obtenção da μc-rule, não vai ao encontro dos resultados das figuras anteriores para custos quadráticos.

Novamente usando a μc-rule, verifica-se que fazendo uma iteração no processo, ou seja, no caso de chegarem clientes às filas se tem:

$$\begin{cases} a_1\mu_1 x_1 - a_1\mu_1 (x_1 - 1) = a_1\mu_1 = \mu_1 C_1 \\ a_2\mu_2 x_2 - a_2\mu_2 (x_2 - 1) = a_2\mu_2 = \mu_2 C_2 \end{cases}$$

correspondendo estes dois valores finais à variação do custo entre duas iterações para cada uma das classes de clientes. Como referido na secção 2.1.6, a μc-rule estabelece prioridades de acordo com o valor de $\mu_i c_i$ sendo a classe portadora do maior produto a prioritária, ou seja, se $\mu_1 C_1 > \mu_2 C_2$ então deverá ser dada prioridade à classe 1. Pode então concluir-se que a prioridade desta classe é atribuída de acordo com a variação do custo entre duas iterações.

Fazendo agora a analogia com o exposto acima para o caso dos novos critérios de custo adoptados:

$$\begin{cases} \mu_1 \left[a_1 x_1 + b_1 x_1^2\right] - \mu_1 \left[a_1 (x_1 - 1) + b_1 (x_1 - 1)^2\right] = \mu_1 \left[a_1 + 2x_1 b_1 - b_1\right] = \mu_1 \Delta_1 \\ \mu_2 \left[a_2 x_2 + b_2 x_2^2\right] - \mu_2 \left[a_2 (x_2 - 1) + b_2 (x_2 - 1)^2\right] = \mu_2 \left[a_2 + 2x_2 b_2 - b_2\right] = \mu_2 \Delta_2 \end{cases}$$

onde neste caso que $\mu_i \Delta_i$ corresponde à diferença entre duas iterações com o novo critério de custo. Novamente optando por atribuir maior prioridade à classe com maior $\mu_i \Delta_i$ deverá ter-se que:

$$\begin{aligned} &\textit{if } \mu_1 \Delta_1(t) > \mu_2 \Delta_2(t) \\ &y(t) = 1 \\ &\textit{if } \mu_1 \Delta_1(t) < \mu_2 \Delta_2(t) \\ &y(t) = 2 \end{aligned}$$

Partindo do pressuposto anterior é então possível encontrar uma expressão que relaciona a variação do custo das duas classes bem como estabelecer prioridades para cada classe dependendo dos parâmetros escolhidos:

$$\begin{aligned} &\mu_1 \left[a_1 + 2x_1 b_1 - b_1\right] = \mu_2 \left[a_2 + 2x_2 b_2 - b_2\right] \Longleftrightarrow \\ &x_2 = \tfrac{(a_1 \mu_1 - a_2 \mu_2) + (b_2 \mu_2 - b_1 \mu_1)}{2 b_2 \mu_2} + \tfrac{b_1 \mu_1}{b_2 \mu_2} x_1. \end{aligned}$$

A expressão anterior é válida para $b_1 \neq 0$ e $b_2 \neq 0$, correspondendo a mesma ao *threshold* que determina a mudança de prioridade das classes. Graficamente o resultado genérico da política para este caso concreto é ilustrado pela Figura 5.10.

No caso em que $b_1 = 0$, tem-se:

$$\begin{aligned} &\mu_1 a_1 = \mu_2 \left[a_2 + 2x_2 b_2 - b_2\right] \Longleftrightarrow \\ &x_2 = \tfrac{(a_1 \mu_1 - a_2 \mu_2) + (b_2 \mu_2)}{2 b_2 \mu_2}, \end{aligned}$$

e, para $b_2 = 0$:

$$\begin{aligned} &\mu_2 a_2 = \mu_1 \left[a_1 + 2x_1 b_1 - b_1\right] \Longleftrightarrow \\ &x_1 = \tfrac{(a_2 \mu_2 - a_1 \mu_1) + (b_1 \mu_1)}{2 b_1 \mu_1}. \end{aligned}$$

Para estes dois últimos casos, o *threshold* é definido por uma recta que poderá ter declive nulo ou infinito, podendo dizer-se que se tem nestes casos uma política de *threshold*. Por exemplo, se $b_1 = 0$

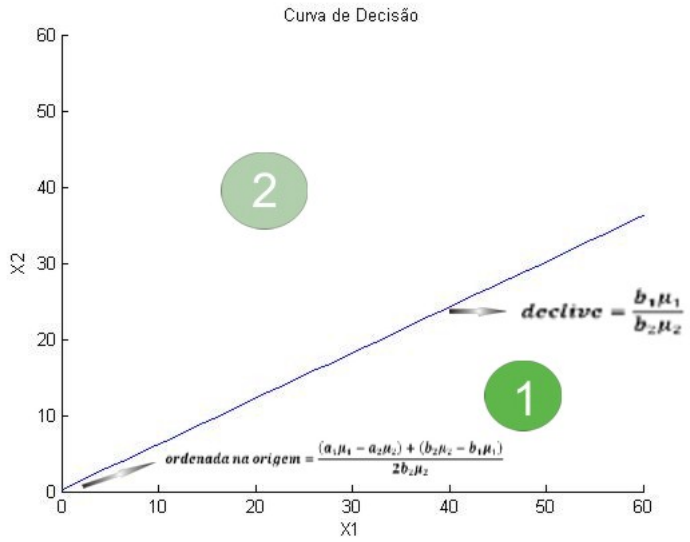


Figura 5.10: Matriz de decisão genérica da $\mu\Delta - rule$.

e $b_2 \neq 0$, existe um valor k, acima do qual a classe 2 tem maior prioridade do que a classe 1. É ainda possível fazer uma correspondência entre o *threshold* de prioridade e o tempo de espera de cada classe, [19]. Se no buffer da classe não prioritária existirem k clientes, conclui-se que o primeiro cliente a chegar ao buffer já esteve à espera para ser servido durante k intervalos de tempo onde cada um destes intervalos corresponde ao intervalo de tempo entre a chegada de dois clientes.

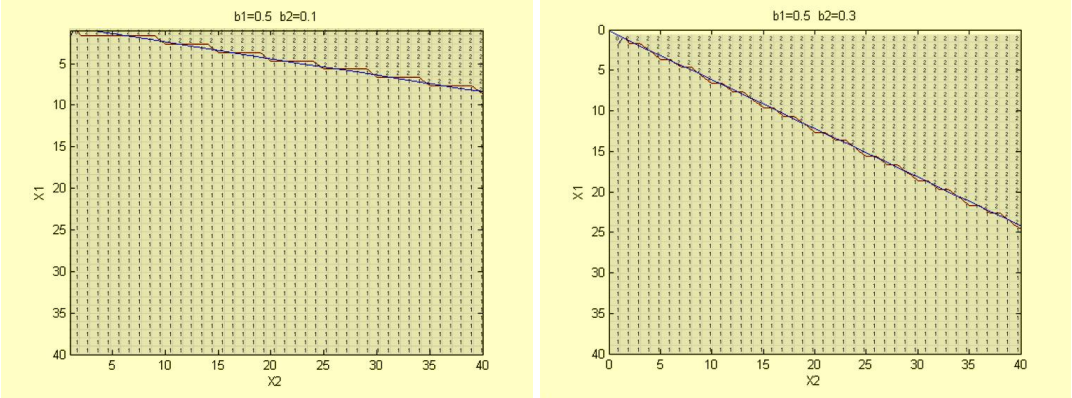
Finalmente, sobrepondo os resultados obtidos com a PD com o resultado obtido pela formulação teórica, obtêm-se os gráficos da Figura 5.11. Estes resultados correspondem a uma validação da formulação intuitiva através dos métodos numéricos uma vez que os dois resultados são coincidentes. Ou seja, a recta que determina o *threshold* de decisão intuitivo sobrepõe-se ao *threshold* obtido a partir das matrizes de decisão.

5.2.3 Variância do Tempo de ciclo

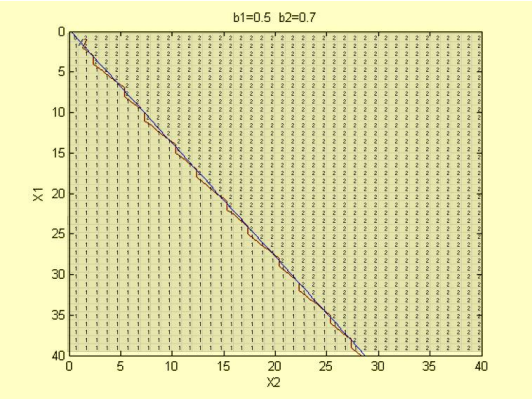
Resta agora verificar se esta política realmente faz jus aos objectivos inicialmente considerados. Ou seja, se esta política não tem um comportamento estrito no tratamento dos clientes. Para verificar esta propriedade simulou-se a rede da Figura 3.1 com os seguintes parâmetros:

$$\begin{cases} \lambda_1 = 0.475, \mu_1 = 1, a_1 = 2, b_1 = 0 \\ \lambda_2 = 0.475, \mu_2 = 1, a_2 = 1, b_2 = ? \\ \beta = 0.002 \end{cases}$$

Neste caso optou-se por simular redes com os parâmetros de custo lineares e quadráticos, fazendo variar o termo quadrático da classe com menor prioridade, b_2, de modo a que fosse visível o efeito



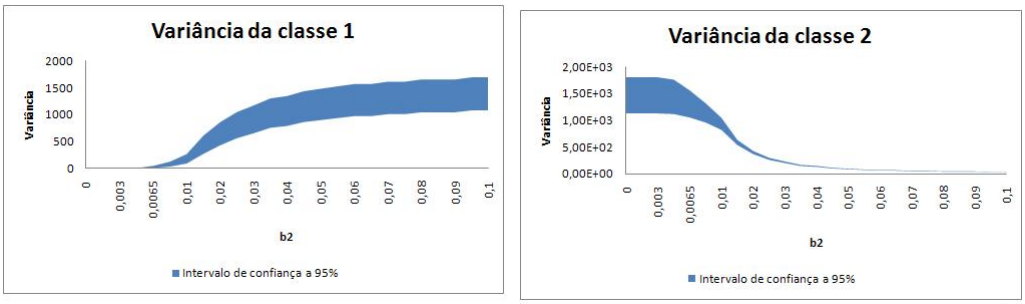
(a) (b)



(c)

Figura 5.11: Matrizes de decisão e resultado teórico.

do aumento deste termo nas variâncias do tempo de ciclo das duas classes. No que diz respeito ao processamento pós simulação, neste caso foi necessário calcular a média das variâncias dos tempos de ciclo para todas as simulações realizadas para um dado conjunto de parâmetros e posteriormente calcular o respectivo intervalo de confiança a 95%.



(a) (b)

Figura 5.12: Sobreposição dos custos obtidos por PD no intervalo de confiança.

Como pode ser verificado, na Figura 5.12, com o aumento de b_2 a variância do tempo de ciclo da classe prioritária aumenta enquanto que a variância da classe não prioritária diminui. Estes resultados

correspondem à expectativa inicial uma vez que com o ajuste do parâmetro b_2 é possível dar mais equidade às duas classes no que diz respeito à sua prioridade.

5.2.4 Variância Global e Dependência dos Processos

Obtida a nova política e analisados os resultados de cada classe individualmente resta agora especular acerca do efeito desta nova política na variância global do tempo de ciclo do sistema. Ainda no que diz respeito à performance da política pode pôr-se a seguinte questão: existirá algum conjunto de parâmetros que usados na nova política conduzam a um valor de variância global do tempo de ciclo mais baixo do que usando a μc-rule com o mesmo conjunto de parâmetros lineares?

Para responder a esta questão, processaram-se ficheiros obtidos pela simulação desta vez para calcular a variância global do tempo de ciclo do sistema bem como o intervalo de confiança a 95%.

Figura 5.13: Variância do tempo de ciclo global.

Como pode ser visto pelo gráfico da figura 5.13, existem de facto alguns valores de b_2 para os quais a variância global do tempo de ciclo é inferior à variância global obtida com os mesmos parâmetros lineares. O valor só com parâmetros lineares corresponde a $b_2 = 0$.

Por fim, é também possível verificar se as variâncias dos tempos de ciclo das classes são independentes entre si. Tal pode ser verificado, comparando a soma das duas variâncias das classes com o resultado obtido quando é calculada a variância global por si só.

Como seria previsível, os processos não são independentes, como pode ser verificado pela Figura 5.14. Uma vez que as classes competem pelo mesmo recurso, se este está a atender um cliente de uma classe, mesmo que este não seja prioritário uma vez que o serviço é *non-preemptive*, os clientes das classes prioritárias que entretanto estiverem à espera para ser processados terão de esperar pelo fim de serviço. Por outro lado, os clientes das classes não prioritárias estão dependentes dos prioritários.

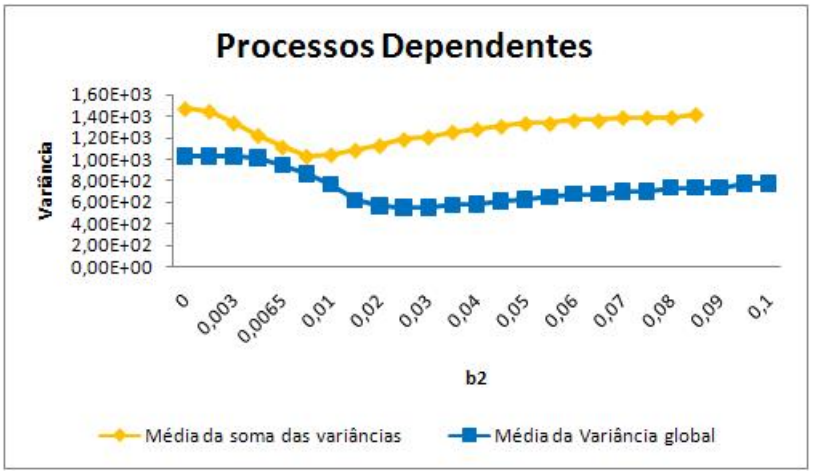


Figura 5.14: Dependência da variância do tempo de ciclo da classes de clientes.

Capítulo 6

Conclusões e Desenvolvimentos Futuros

A nível geral, duas conclusões podem ser retiradas dos resultados anteriormente apresentados:

- a modelação de redes de filas de espera para resolução de alguns problemas usando apenas uma o conteúdo dos buffers para representar o estado não corresponde a uma formulação correcta;
- formulou-se uma função de custo que produz um política de sequenciamento alternativa às políticas de prioridade estrita que, relativamente a estas, reduz a variância do tempo de ciclo.

Decorrente da primeira conclusão, pode especular-se que poderão existir políticas de decisão mal formuladas em consequência de uma errada modelação das redes usada na formulação dos problemas. Ou seja, poderão existir problemas que tenham sido solucionados fazendo uso de modelos incorrectos e os resultados serão, em consequência, incorrectos. Analisando os resultados pode verificar-se que, apesar de o modelo a uma variável produzir estimativas incorrectas do custo, a política que se obtém por aplicação do algoritmo de iteração de valor é a μc-rule, [2], que se sabe estar correcta para o problema em que os custos são lineares como função do número de clientes no sistema. No entanto, pode-se conjecturar que, em geral para outros PDM's produzidos no contexto das filas de espera que apenas façam uso do número de clientes para representar o estado, poderão haver erros no próprio cálculo das políticas de decisão.

Deve ter-se em conta que poderão existir problemas em que as duas formulações conduzam a duas políticas com comportamentos diferentes estando apenas uma das soluções correcta. Tal deve-se a situações em que as transições não controláveis e não conducentes a pontos de decisão permitem o cruzamento de fronteiras separando regiões onde a decisão óptima é diferente.

Ainda a ter em conta que a formulação tradicional de problemas de decisão de Markov não contempla a existência de transições imediatas derivadas da escolha de uma acção. Entende-se que não há duas transições simultâneas e, por isso, o estado em que se permanece após uma decisão enquanto não houver transição é o mesmo estado anterior a uma decisão. Na formulação que se apresentou nesta dissertação, para capturar com rigor a dinâmica da rede em estudo, foi necessário incluir uma transição instantânea como resultado da tomada de decisão. Assim, na ausência de transição posterior, o estado em que o sistema permanece não é o mesmo que o estado pré-decisão. Outra coisa não poderia ser, dado que após uma transição que coloca um sistema num ponto de decisão, é tomada de facto uma decisão. Instantes futuros, imediatamente após a decisão tomada e onde não ocorra outra transição, não podem ser considerados como pontos de decisão. Daí a necessidade de se definir um estado pré-decisão e outro pós-decisão. Por exemplo, um ponto de decisão para o estado $[x_1, x_2, 0]$, com prioridade à classe 1 e com $x_1 \neq 0$ tem aquele como estado pré-decisão e tem o estado $[x_1, x_2, 1]$ como estado pós-decisão.

Em consequência deste facto, pode sugerir-se que a resolução deste tipo de problemas usando técnicas de programação dinâmica deverá ser reformulada de modo a que resultados correctos sejam obtidos. Nomeadamente no que concerne ao estabelecimento formal de propriedades das políticas óptimas. Este aspecto não está contemplado na literatura relativa a Programação Dinâmica e PDM's e necessita ser endereçado.

No que diz respeito à formulação da nova política, pode verificar-se que o objectivo inicial foi atingido com sucesso. A política formulada permite, através da manipulação dos parâmetros dar mais ou menos prioridade às classes não prioritárias de acordo com o desejado. Em caso extremo, anulando as componentes quadráticas, a política coincide com a μc-rule, o que não deixa de ser uma consequência lógica, uma vez que esta nova política é uma variante da μc-rule.

Pode ainda concluir-se que, por observação dos resultados decorrentes da formulação intuitiva de política, que a derivação de políticas usando funções genéricas monotónicas não decrescentes pode ser válida. Intui-se então que a formulação de uma nova política poderá ser derivada da variação de uma iteração do processo de uma função genérica monotónica não decrescentes multiplicada pela taxa de processamento. Ou seja, se $g(x_1, x_2) = g_1(x_1) + g_2(x_2)$, $\Delta_i = g_i(x_i) - g_i(x_i - 1)$, para os casos em que $g_i(x_i + \delta) \geq g_i(x_1)$ para $\delta > 0$. Naturalmente que, apesar de se ter considerado nesta dissertação apenas duas filas, o resultado que aqui é apresentado é generalizável para mais que duas classes de clientes.

Quanto ao desempenho da política, foi observada a variância do tempo de ciclo de cada uma das classes da rede bem como a variância do tempo de ciclo global. No caso em que os parâmetros quadráticos não têm relevância, ou seja, quando a política é igual à μc-rule, a variância do tempo de espera

das classes não prioritárias é muito maior do que os das classes prioritárias, como seria de esperar. Quando se aumentam os parâmetros quadráticos de modo a favorecer as classes menos prioritárias, a variância das classes prioritárias aumenta enquanto que a das não prioritárias diminui. Quanto à variância do tempo de espera global, pode concluir-se que esta política tem melhor desempenho do que a μc-rule, uma vez que existe um conjunto de parâmetros que permite a obtenção de uma variância global mais baixa.

É ainda possível relacionar-se estes resultados com o tempo de espera para os clientes. Quando se tem em conta a μc-rule, dizer que a função de custo é linear é, em certo sentido, dizer que a "paciência" de um cliente que esteja à espera durante por exemplo meia hora ou há três horas é a mesma. Ou seja, a "paciência" marginal de esperar é constante. Já no caso da nova função de custo, pode verificar-se que o mesmo não acontece. Ou seja, a "paciência" de um cliente vai-se degradando à medida que o tempo de espera vai aumentando. Ou seja, a "paciência" marginal degrada-se linearmente com o tempo. Daí o custo quadrático. O primeiro cliente de cada fila tem um tempo de espera por serviço igual à soma dos intervalos entre chegadas de clientes da sua classe. Por isso, um maior tamanho na fila corresponde a um maior tempo de espera, em valor médio.

Com a componente quadrática na função de custo pode constatar-se que a prioridade de uma classes aumenta à medida que o número de clientes dessa mesma classe aumenta, que é o mesmo que dizer que para cada escolha da componente quadrática da classe não prioritária se está implicitamente a definir qual o tempo aceitável de espera para essa classe.

6.1 Desenvolvimentos Futuros

Quando falamos do problema da modelação de uma rede de filas de espera, uma vez que foi verificado que a modelação de alguns tipos de redes deve ser realizada tendo em conta o estado dos servidores presentes na rede, resta fazer uma abordagem mais generalista no sentido de caracterizar os sistemas que realmente necessitam de ser modelados de acordo com estas especificações. Um outro aspecto seria no sentido de verificar se realmente existe a possibilidade de haver políticas mal formuladas, ou seja, verificar a existência de casos em que um mesmo problema possa conduzir a duas políticas diferentes decorrente do modo como a rede é modelada e verificar qual dos modelos corresponde ao correcto.

No que diz respeito às transições imediatas, um desenvolvimento necessário consiste na reformulação dos problemas de decisão de Markov fazendo uso de programação dinâmica que contemplem esta particularidade.

No caso da política aqui proposta, o próximo passo consiste na demonstração formal da optimalidade da estrutura intuída.

Apêndice A

Unified Modelling Language do simulador.

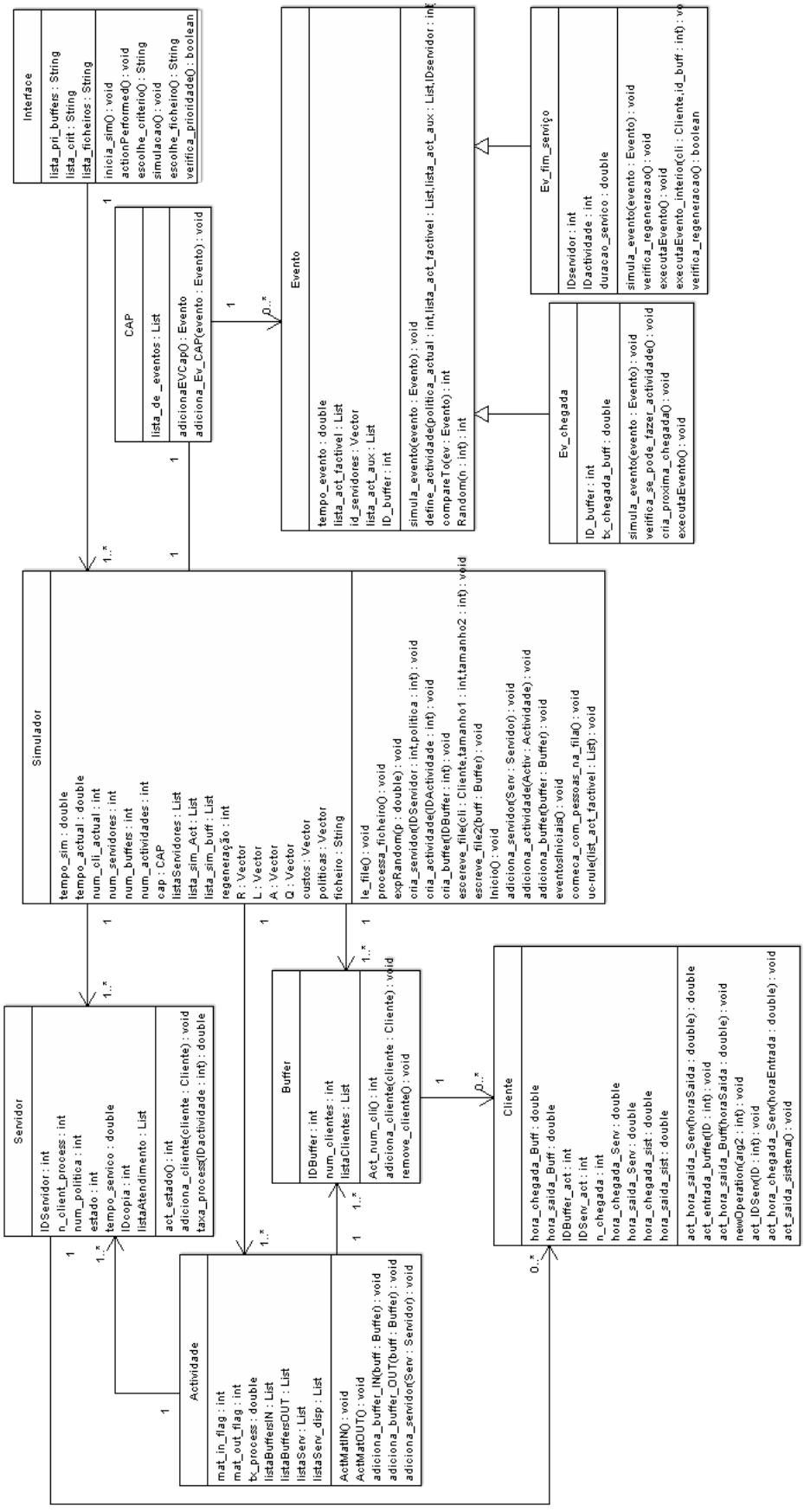


Figura A.1: UML.

Bibliografia

[1] K. Arnold, J.Gosling, and D. Holmes. *Java Programming Language*. Addison Wesley, fourth edition, 2005.

[2] J. S. Baras, A. J. Dorsey, and Makowski. Two competing queues with linear costs: The μc-rule is often optimal. *Advances in Applied Probability*, 17(1):186–209, 1985.

[3] R. Bellman. On the theory of dynamic programming. *Proceedings of The National Academy of Sciences*, 38:716–719, 1952.

[4] R. Bellman. An introduction to the theory of dynamic programming. *The Rand Corporation, Santa Monica California*, 1953.

[5] D. P. Bertsekas. *Dynamic Programming and Optimal Control, Volume One*. Athena Scientific, 1995.

[6] D. P. Bertsekas. *Dynamic Programming and Optimal Control, Volume Two*. Athena Scientific, 1995.

[7] C. Buyukkoc, P. Varaiya, and J. Walrand. The μc-rule revisited. *Advances in Applied Probability*, 17(1):237–238, 1985.

[8] C. G. Cassandras. *Discrete Event Systems: Modeling and Performance Analysis*. Asken Associates Incorporated Publishers, 1993.

[9] A. Cobham. Priority assignment in waiting line problems. *Operations Research*, 2:70–76, 1954.

[10] D. R. Cox and W. L. Smith. *Queues*. Chapman and Hall, 1961.

[11] J. M. Harrison. Dynamic scheduling of a multiclass queue: Discount optimality. *Operations Research*, 23(2):270–282, 1975.

[12] J. M. Harrison. A priority queue with discounted linear costs. *Operations Research*, 23(2):260–269, 1975.

[13] J. M. Harrison. Brownian models of open processing networks: Canonical representation of workload. *The Annals of Applied Probability*, 10(1):75–103, 2000.

[14] J. M. Harrison. Stochastic networks and activity analysis. In *Analytic Methods in Applied Probability*, pages 53–76. American Mathematical Society, 2002.

[15] G. M. Koole. Assigning a single server to inhomogeneous queues with switching costs. *Theoretical Computer Sceince*, 182:203–216, 1997.

[16] G. M. Koole and P. Nain. On the value of a priority queue with an application to a controlled polling model. *Queuing Systems*, 34:199–214, 2000.

[17] S. A. Lippman. Applying a new device in the optimization of exponential queuing systems. *Operations Research*, 23:687–710, 1975.

[18] M. Rebello. Simulação de redes de actividades em JAVA. Aplicação à estabilização de redes de filas de espera. Master's thesis, IST, Outubro 2007.

[19] M. Rebello, Z. Fernandes, and C. Bispo. Simulating activity networks in JAVA. 6th Eurosim Congress on Modelling and Simulation, 2007.

[20] S. M. Ross. *Stochastic Processes*. John Willey & Sons, 1996.

[21] J. Walrand. *An Introduction to Queuing Networks*. Prentice Hall, 1988.