UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE MATEMÁTICA
DEPARTAMENTO DE ESTATÍSTICA

O uso da simulação de eventos discretos na avaliação de tecnologias em saúde

Autor: Mauro Lacerda
Orientador: Prof.ª Dra. Patrícia Klarmann Ziegelmann

Porto Alegre, 06 de Dezembro de 2011.

Universidade Federal do Rio Grande do Sul
Instituto de Matemática
Departamento de Estatística

O uso da simulação de eventos discretos na avaliação de tecnologias em saúde

Autor: Mauro Lacerda

Monografia apresentada para obtenção
do grau de Bacharel em Estatística.

Banca Examinadora:

Prof.ª Dra. Patrícia Klarmann Ziegelmann (orientadora)
Me. André Luís Ferreira da Silva

Porto Alegre, 06 de Dezembro de 2011.

“Hipótese é algo que não é,

mas a gente finge que é,

só para sabermos como seria se ela fosse!”

(autor desconhecido)

Agradecimentos

Agradeço aos meus pais, Gilmar Rodrigues de Lacerda e Marisa Maria Girotto Lacerda, e a minha irmã, Tamara Lacerda, por terem me incentivado a iniciar esse curso de graduação e também por todo o apoio que sempre me deram.

Agradeço à minha namorada, Mariana Bartels, por toda a ajuda que me deu durante o curso, por ter me incentivado a estudar, mesmo sem querer, e principalmente por todos os momentos felizes que passamos juntos.

Agradeço também à professora Patrícia Klarmann Ziegelmann por ter aceitado me orientar neste trabalho e por todas as ajudas que me deu durante a elaboração deste.

Resumo

As tecnologias em saúde referem-se às ferramentas que visam alcançar algum diagnóstico ou propósito terapêutico. Como os recursos utilizados na área da saúde são escassos, é preciso decidir qual é o melhor caminho de ação a ser tomado. A análise comparativa dos diferentes caminhos de ação em termos de seus custos e de suas consequências é o que define a avaliação econômica das tecnologias em saúde.

O propósito da modelagem na avaliação econômica de tecnologias em saúde é estruturar a evidência dos efeitos clínicos e econômicos de uma maneira que possa ajudar na tomada de decisões sobre a alocação de recursos de saúde. Existem diversas técnicas que podem ser utilizadas na construção de um modelo com esse propósito. As árvores de decisão e os modelos de Markov são as duas técnicas mais amplamente utilizadas na avaliação de tecnologias em saúde, mas que possuem certas limitações que impedem que sejam utilizadas para criar modelos capazes de representar todos os problemas de decisão que podem ser vistos na prática. A simulação de eventos discretos ainda é uma técnica pouco utilizada, mas que pode apresentar diversas vantagens em relação às outras, principalmente em situações que envolvam elevado grau de complexidade, além de ser extremamente flexível, sendo indicada principalmente para descrever situações que envolvam interação entre pacientes.

Este trabalho apresenta uma breve descrição das árvores de decisão e dos modelos de Markov, bem como um exemplo que descreve detalhadamente a construção de um modelo de eventos discretos com o objetivo de torná-lo facilmente reprodutível, na esperança de que este seja útil e que possa servir como base para quem desejar criar um modelo utilizando esta mesma técnica. Para a construção desse exemplo será utilizado o *software* MATLAB. Esse exemplo demonstra a aplicabilidade da simulação de eventos discretos para avaliar situações que envolvam dependência ou interação entre pacientes, como em situações em que existe uma fila de espera para os pacientes receberem um atendimento. Por fim, através deste mesmo exemplo são evidenciadas as diferenças que podem ocorrer nos resultados de uma simulação por causa da escolha incorreta da técnica utilizada na construção do modelo.

Abstract

The healthcare technologies refer to the tools that aim to achieve a diagnostic or therapeutic purpose. As the resources used in health care are scarce, it must be decided what is the best course of action to be taken. The comparative analysis of different courses of action in terms of its costs and consequences is what defines the economic evaluation of healthcare technologies.

The purpose of modeling in the economic evaluation of health technologies is to structure the evidence of clinical and economic effects in a way that can help to make decisions about the allocation of health resources. Several techniques can be used to build a model for this purpose. Decision trees and Markov models are the two techniques most widely used in the evaluation of healthcare technologies, but they have certain limitations that prevent them from being used to create models capable of representing all decision problems that can arise in practice. The discrete event simulation is still an underused technique, although it can present several advantages over the others, especially in situations involving a high degree of complexity, in addition to being extremely flexible and especially suitable to describe situations that involve interaction between patients.

This work presents a brief description of decision trees and Markov models, as well as an example that fully describes the construction of a discrete event model in order to make it easily reproducible, hoping that it can be helpful and used as a basis for those who wish to create a model using the same technique. The MATLAB software will be used to make this example. This example demonstrates the applicability of discrete event simulation to evaluate situations involving dependence or interaction between patients, as in situations where there is a waiting list for patients receiving a treatment. Finally, through this same example the differences that may occur in the results of a simulation because of the incorrect choice of the technique used in constructing the model are highlighted.

Sumário

1. Introdução

Todos os dias inúmeras pesquisas são realizadas nas mais diversas áreas do conhecimento, resultando na criação de novas tecnologias ou em melhorias daquelas já existentes. O termo tecnologia é utilizado para se referir às ferramentas, aos métodos e aos materiais científicos empregados para alcançar um determinado objetivo. Na avaliação de tecnologias em saúde o termo refere-se às ferramentas que visam alcançar algum diagnóstico ou propósito terapêutico (STAHL, 2008).

Como os recursos utilizados na área da saúde são escassos, é preciso identificar e medir os efeitos e os custos associados a essas tecnologias para que seja possível avaliar e decidir qual é a melhor decisão a ser tomada. Essa análise comparativa dos diferentes caminhos de ação em termos de seus custos e de suas consequências é o que define a avaliação econômica das tecnologias em saúde (KARNON e BROWN, 1998).

A avaliação econômica é realizada a fim de informar sobre o uso adequado de determinadas intervenções ou programas de saúde para a tomada de decisões. Para avaliar os custos e benefícios de novas tecnologias é necessário que seja feita uma síntese das evidências existentes. Os ensaios clínicos randomizados são o tipo de estudo mais recomendado para se comparar o efeito de diferentes tecnologias, mas geralmente não envolvem nenhuma avaliação econômica. Além disso, ensaios clínicos podem até possuir grande validade interna, mas geralmente englobam um espaço curto de tempo e apenas uma limitada parte das opções disponíveis (CARO, MÖLLER e GETSIOS, 2010). O uso das técnicas de modelagem de decisão é uma alternativa para essas situações em que os ensaios clínicos executados não incluem as informações econômicas necessárias, ou quando esses ensaios clínicos ainda não foram ou não podem ser conduzidos (KARNON e BROWN, 1998).

O propósito da modelagem é estruturar a evidência dos efeitos clínicos e econômicos de uma maneira que possa ajudar na tomada de decisões sobre a alocação de recursos de saúde. Obtendo um claro entendimento da relação entre a efetividade e o custo de diferentes estratégias é possível avaliar a custo-efetividade e determinar quais as opções que devem ser adotadas com base nas informações disponíveis (PHILIPS *et al.*, 2006; WEINSTEIN *et al.*, 2003).

Para construir um modelo é necessário agrupar a melhor informação e conhecimento disponível sobre a estratégia estudada, identificar quais são os resultados de interesse, bem como os riscos, taxas e probabilidades que afetam cada ação. Isso faz com que os modelos combinem evidências obtidas através de várias fontes, incluindo dados de ensaios clínicos, estudos observacionais, opiniões de especialistas, entre outras (STAHL, 2008; WEINSTEIN *et al.*, 2003). Também é necessário estruturar as possíveis trajetórias dos pacientes de uma maneira lógica e realista através do tempo, levando em consideração os eventos que podem ocorrer e suas implicações clínicas e econômicas (LE LAY *et al.*, 2006).

A estrutura desses modelos não é especificada pelos tomadores de decisão. Ela é escolhida pelo analista para que reflita da melhor maneira possível as necessidades do problema em questão, normalmente levando em consideração a relação entre os dados que se possui e os resultados que são de interesse. Se uma estrutura incorreta é utilizada os resultados do modelo poderão ser falhos e a decisão tomada poderá também ser incorreta (BRENNAN, CHICK e DAVIES, 2006). Assim, para serem ferramentas úteis, os modelos devem refletir razoavelmente bem o que se entende sobre o problema e, embora sempre envolvam simplificação da realidade, é importante que essas simplificações não a distorçam de maneira que os resultados obtidos sejam errôneos (CARO, MÖLLER e GETSIOS, 2010).

O valor do modelo não está somente nos resultados que ele gera, mas também em sua habilidade de revelar a ligação entre os dados que se possui e os resultados obtidos na forma de efeitos e custos. A falta de entendimento sobre um modelo e a falta de confiança em seus resultados podem limitar o quanto a informação gerada será considerada. Por esse motivo, o modelo não pode ser uma caixa-preta para o usuário final e sim ser o mais transparente possível, de modo que a lógica por trás dos resultados possa ser compreendida de maneira intuitiva. Também por essa razão, os resultados obtidos através do modelo não devem ser apresentados somente por meio de estimativas pontuais ou de alegações incondicionais de custo e efetividade, e deve ser realizada uma ampla análise da sensibilidade dos resultados em relação a diferentes suposições ou dados incorporados ao modelo (VAN GESTEL *et al.*, 2010; WEINSTEIN *et al.*, 2003).

Deve-se lembrar de que a principal motivação para utilizar a modelagem na avaliação de problemas na área da saúde é proporcionar uma ferramenta capaz de ajudar pacientes, de ajudar os sistemas de saúde a agir com maior eficiência, e de proporcionar alto valor e qualidade nos serviços de saúde (FENDRICK, 2006).

Se bem executada, a modelagem na avaliação de custo efetividade pode ajudar a informar e possivelmente persuadir os tomadores de decisão a tomar a melhor decisão possível, produzindo rapidamente um grande retorno do valor investido, apesar de custar uma parte muito pequena do orçamento total de um determinado projeto (STAHL, 2008).

Existem diversas técnicas que podem ser utilizadas na construção de um modelo para realizar a avaliação econômica de uma tecnologia em saúde. A mais simples destas técnicas é a árvore de decisão, que é basicamente a representação gráfica de um problema de decisão. Através dela são representadas as possíveis alternativas para um problema, juntamente com o desenrolar de cada uma destas alternativas.

Outra técnica que pode ser utilizada é o modelo de Markov, que tem sido a mais utilizada em avaliações de tecnologias em saúde. O modelo de Markov trabalha com a representação dos estados de saúde dos pacientes, e permite que sejam representadas facilmente situações que envolvam eventos recorrentes ou longos períodos de tempo. Porém, algumas limitações dos modelos de Markov, principalmente a incapacidade de incorporar a interação entre os pacientes, abrem espaço para a utilização da simulação de eventos discretos, que surge como uma alternativa para a criação de modelos capazes de representar situações com maior grau de complexidade, que não poderiam ser representadas por um modelo de Markov.

Assim, este trabalho foi criado com o propósito de comparar a simulação de eventos discretos com as árvores de decisão e os modelos de Markov. Por este motivo, é feita uma breve descrição destas duas técnicas, bem como uma descrição mais detalhada da simulação de eventos discretos. Além disso, é feita a comparação das principais características de cada uma delas na tentativa de tornar mais fácil a escolha da técnica correta de acordo com a situação que será estudada.

Por ser ainda uma técnica pouco conhecida e utilizada, é demonstrado um passo a passo para o desenvolvimento de um modelo para a simulação de eventos discretos utilizando o *software* MATLAB®, bem como a análise de seus resultados.

No capítulo 2 deste trabalho são vistas as principais características das árvores de decisão, dos modelos de Markov, e da simulação de eventos discretos. No capítulo 3 é realizada a comparação destas três técnicas. Por fim, no capítulo 4 demonstra-se a construção de um modelo de eventos discretos e a análise dos resultados obtidos.

2. Técnicas de Modelagem

Neste capítulo são abordadas três das principais técnicas de simulação utilizadas na análise de custo-efetividade de tecnologias em saúde. É descrito o funcionamento e as principais características das árvores de decisão, dos modelos de Markov, e da simulação de eventos discretos, sendo essa última vista mais detalhadamente, além de uma breve demonstração da maneira como é construído um modelo utilizando árvores de decisão e modelos de Markov. A construção de um modelo para a simulação de eventos discretos é demonstrada no capítulo 4.

Essas técnicas podem ser empregadas no desenvolvimento de modelos nas mais diversas áreas do conhecimento, como engenharia de produção ou economia, por exemplo. Neste trabalho, a linguagem e os exemplos utilizados referem-se à área da saúde.

2.1. Árvores de Decisão

Uma árvore de decisão é fundamentalmente a representação gráfica de um problema de decisão. De todas as técnicas de modelagem utilizadas em avaliação de tecnologias em saúde, é considerada a mais simples. Serve como uma ferramenta capaz de estruturar e combinar as informações obtidas por meio de diversas fontes disponíveis, podendo ajudar o tomador de decisões na melhor compreensão do problema em questão.

O modelo de árvore de decisão é criado a partir de um gráfico simples de uma estrutura ramificada, contendo diversos caminhos que representam as possíveis decisões a serem tomadas. O gráfico engloba todas as possíveis alternativas para o problema de decisão, bem como a sequência dos eventos que podem ocorrer em virtude da escolha de cada uma dessas alternativas. A Figura 1 mostra a estrutura de uma árvore de decisão.

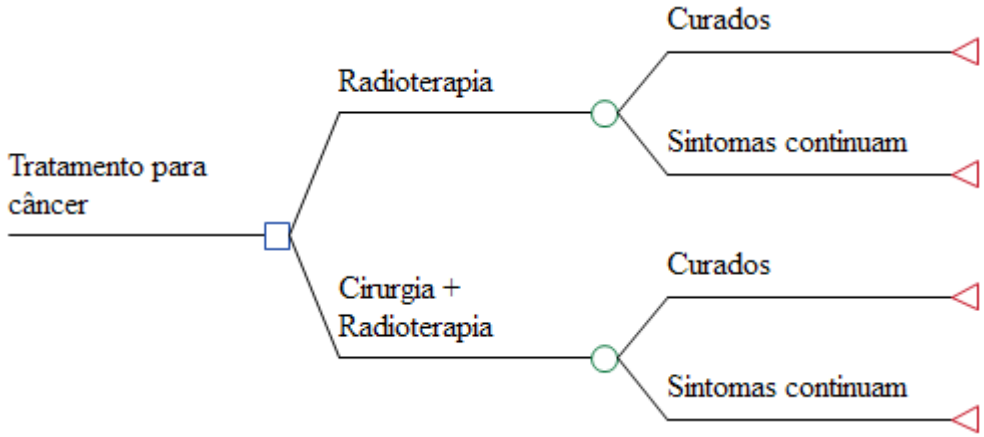

Figura 1 – Exemplo de uma árvore de decisão para o tratamento de um câncer.

A principal vantagem para se utilizar um modelo de árvore de decisão é a facilidade com que este é desenvolvido em comparação às outras técnicas, servindo muito bem para grande parte dos problemas de decisão. Além disso, os modelos criados são, em geral, bastante transparentes, no sentido de que é fácil compreender o que foi feito e transmitem facilmente as informações utilizadas, bem como os resultados obtidos.

O desenvolvimento de um modelo de árvore de decisão é usualmente dividido em três etapas principais: (1) a construção da estrutura da árvore, (2) o preenchimento da árvore com as probabilidades associadas a cada ramo e as medidas de custo e efetividade, e (3) a resolução da árvore (*roll-back*) para encontrar os custos e a efetividade associados a cada uma das decisões.

2.1.1. Construindo uma árvore de decisão

A primeira etapa na construção de uma árvore de decisão envolve a criação de sua estrutura, tendo por objetivo representar corretamente todas as possíveis alternativas para o problema e o desenrolar dos eventos que decorrem de cada escolha. Durante a criação, o analista precisa ter em mente que a estrutura deve representar o desenrolar da história natural da doença e do problema em questão.

A representação gráfica da árvore é composta por dois componentes principais, os ramos e os nós. Os ramos representam os possíveis caminhos que podem ser percorridos pelos pacientes, enquanto os nós representam algum acontecimento no processo de decisão ou no desenrolar do problema, e são classificados como nós de decisão, nós de chance, ou nós terminais.

Os nós de decisão representam o ponto em que é tomada uma decisão entre possíveis alternativas. Costumam ser representados através de um quadrado azul, e geralmente são incluídos no início da árvore, representando a primeira decisão que deve ser feita para definir o restante do caminho para cada alternativa. No exemplo apresentado na Figura 1 um nó de decisão foi incluído no início da árvore para indicar que a Radioterapia e a Cirurgia + Radioterapia são as duas tecnologias que estão sendo comparadas, ou seja, o modelo foi construído para decidir entre essas duas alternativas.

Os nós de chance são utilizados para representar um acontecimento que possui mais do que um possível resultado e que é condicionado a uma probabilidade, como o sucesso ou não de um tratamento, por exemplo. São representados por meio de um círculo verde no

gráfico da árvore, e a cada um dos nós de chance incluídos na árvore pelo menos dois ramos precisam ser incluídos. Estes ramos representam os possíveis resultados, sendo necessário atribuir uma probabilidade a cada ramo.

Por fim, os nós terminais representam os resultados do desfecho principal do estudo. São eles que incluem as medidas finais de custo e de efetividade, e são representados no gráfico por um triângulo vermelho.

Por convenção, a árvore é construída seguindo da esquerda para a direita, geralmente tendo um nó de decisão em seu início, e vários caminhos sendo criados a partir dele por meio de nós de chance, seguindo até os nós terminais representando os desfechos. É importante notar que os caminhos criados precisam ser mutuamente exclusivos, ou seja, um paciente só pode percorrer um dos possíveis caminhos, e que não há a possibilidade de um paciente voltar para um nó anterior ao que ele se encontra.

A quantidade e a sequência dos ramos formados pelos nós de chance podem ser diferentes para cada decisão, como pode ser visto na Figura 2, que mostra os três tipos de nós em uma árvore de decisão.

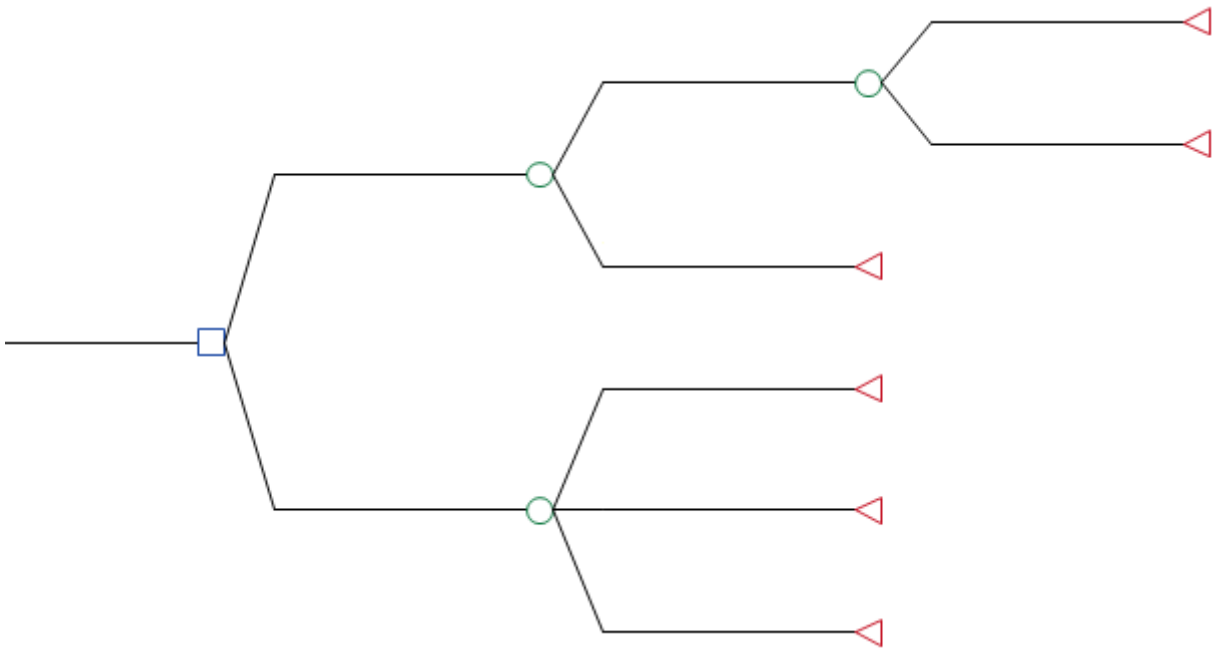

Figura 2 - Árvore de decisão mostrando os três tipos de nós.

2.1.2. Preenchendo a árvore

Quando a estrutura da árvore estiver completa, o próximo passo é preencher a árvore, atribuindo as probabilidades e medidas que são necessárias para a resolução do problema. As informações utilizadas para definir esses valores são obtidas por meio de estudos já publicados ou da opinião de algum especialista na área de estudo, ou seja, são estimativas que possuem certo grau de incerteza.

As probabilidades utilizadas refletem a proporção de pacientes que se direciona para cada um dos possíveis ramos após passar por um nó de chance. A soma das probabilidades associadas a todos os ramos resultantes de um determinado nó de chance deve necessariamente ser igual a um. Esta característica garante que o número de pacientes que entra no modelo seja igual ao que chega até o final, ou seja, trata-se de uma coorte fechada.

Os custos podem ser incluídos diretamente nos nós terminais, de modo a refletir o custo total associado a todos os acontecimentos que compõem o caminho até cada um deles, ou podem ser incluídos aos nós intermediários, refletindo o custo associado ao acontecimento que eles representam. As medidas de efetividade são incluídas nos nós terminais e representam os efeitos percebidos pelos pacientes que percorreram aquele caminho.

2.1.3. Resolvendo (*roll-back*)

A Figura 3 apresenta o exemplo anterior depois de a árvore ter sido preenchida com as probabilidades, custos e efetividades. Neste exemplo, o custo foi medido em reais e a efetividade em anos de vida ajustados pela qualidade. O processo de estimação dos custos e das efetividades médias não será abordado detalhadamente neste trabalho, mas pode ser visto em (DRUMMOND *et al.*, 2005).

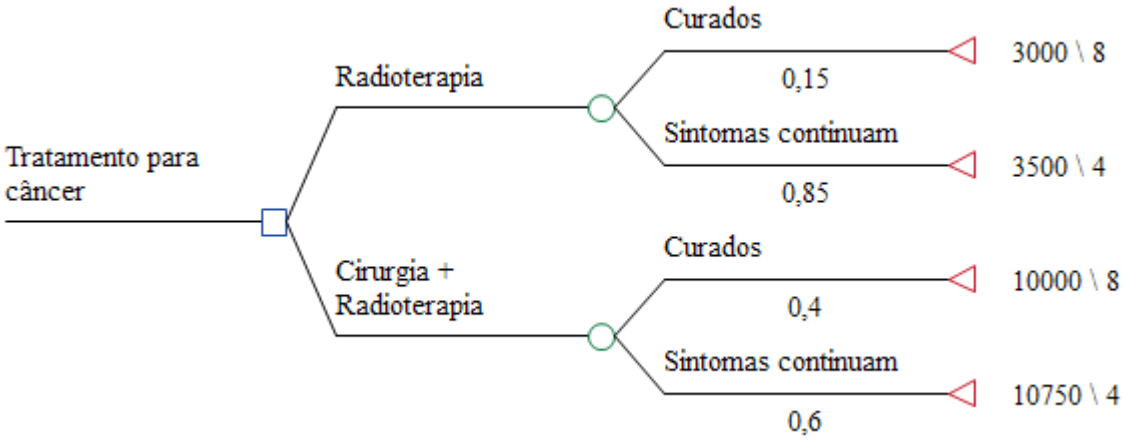

Figura 3 – Exemplo de uma árvore de decisão completa com probabilidades, custos e efetividade.

Resolver a árvore significa calcular os custos e as efetividades esperadas para cada possível decisão. Para obter estes resultados, inicia-se multiplicando as medidas de custo e efetividade associadas aos nós terminais pela probabilidade de ocorrência de cada um deles. Por exemplo, os pacientes que recebem a radioterapia incorrem um custo total de R$ 3000,00, enquanto aqueles que continuam com os sintomas têm um custo total de R$ 3500,00. Agora, a probabilidade de um paciente que recebe a radioterapia ser curado é igual a 0,15 e, portanto, a probabilidade de ele continuar com sintomas é igual a 0,85. Assim, multiplicando cada custo pela sua respectiva probabilidade obtêm-se $0{,}15 * 3000 + 0{,}85 * 3500 = 3425$ reais, que é o custo esperado, ou custo médio, para a radioterapia.

O mesmo processo é executado em todos os outros nós de chance da árvore e também para calcular a efetividade média de cada um dos tratamentos. Depois de obtidas as medidas associadas a cada uma das alternativas é possível escolher qual a melhor decisão a ser tomada.

Esse processo de multiplicação utilizado para obter os resultados é chamado de *roll-back*, devido ao modo que a conta é executada, já que ela é executada a partir do final do modelo e segue até chegar ao início. A Figura 4 mostra os resultados obtidos após o *roll-back* da árvore utilizada neste exemplo.

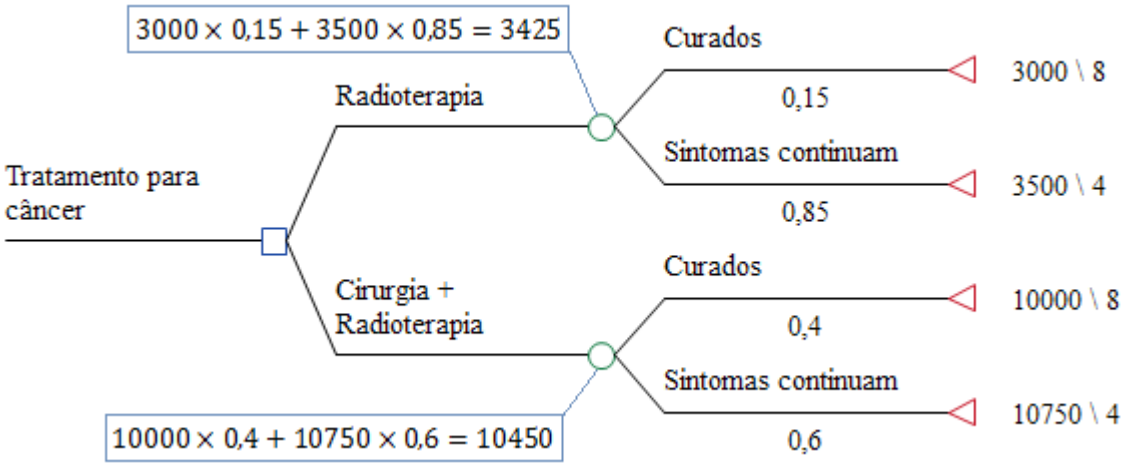

Figura 4 – Exemplo de uma árvore de decisão com resultados de custos.

2.1.4. Limitações dos modelos de árvore de decisão

Apesar de ser uma técnica bastante simples e fácil para desenvolver um modelo, a árvore de decisão possui algumas limitações que fazem com que outras técnicas de modelagem sejam preferidas para alguns tipos de situações.

Em particular, um modelo de árvore de decisão não é capaz de levar em conta a passagem do tempo. Por exemplo, se a probabilidade de um paciente falecer aumenta com cada ano que passa, é preciso criar um novo nó de chance para representar cada ano, o que acaba complicando muito a construção do modelo. Por esse motivo árvores de decisão são indicadas para situações que ocorrem em um curto espaço de tempo.

Outra limitação é o fato de os pacientes somente poderem seguir adiante nos caminhos do modelo, fazendo com que seja difícil a representação de acontecimentos que ocorrem diversas vezes ao longo do tempo avaliado.

Podem ser feitas alterações no modelo para que uma árvore possa representar situações em que tais características sejam necessárias, porém, essa alternativa possui a desvantagem de tornar o modelo demasiadamente complexo. Existem outras técnicas que são mais adequadas para este tipo de situação, como os modelos de Markov, que são descritos a seguir.

2.2. Modelos de Markov

Os modelos de Markov são utilizados para representar processos aleatórios que evoluem com o passar do tempo. São bastante usados quando se deseja trabalhar com situações em que um determinado acontecimento pode ocorrer várias vezes ao longo de um grande período de tempo, devido à facilidade com que estas situações podem ser representadas no modelo.

Em um modelo de Markov, os pacientes são colocados em estados pré-definidos, que costumam representar o estado de saúde do paciente. Esses pacientes podem ir e vir entre os estados, representando as mudanças em seu estado de saúde que ocorrem ao longo do tempo. A Figura 5 mostra uma possível representação para os estados de um modelo de Markov.

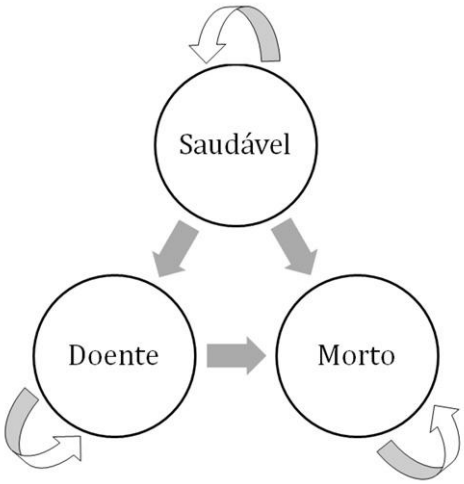

Figura 5 - Diagrama de influência para um modelo de Markov.

Todos os pacientes que estão em um mesmo estado são tratados de maneira homogênea, ou seja, não é feita a diferenciação entre os pacientes em termos do tempo de permanência naquele estado ou do número de vezes que os pacientes passaram pelo estado. Essa é uma característica conhecida como falta de memória dos modelos de Markov.

Ao contrário de uma árvore de decisão, o tempo é parte fundamental dos modelos de Markov e, por isso, são indicados para representar situações em que se deseja avaliar custos e efeitos espalhados em um longo período de tempo.

Além disso, também podem ser utilizados em conjunto com uma árvore de decisão, que é utilizada para determinar a proporção de pacientes que serão avaliados em diferentes modelos de Markov.

A Figura 6 mostra uma cadeia de Markov inserida em uma árvore de decisão utilizada para modelar uma situação em que pacientes com câncer podem ser submetidos a dois tratamentos diferentes. Após o tratamento existe uma probabilidade de os pacientes continuarem apresentando os sintomas da doença, quando então passam a fazer parte de um modelo de Markov que determina o que acontecerá com eles a partir disso.

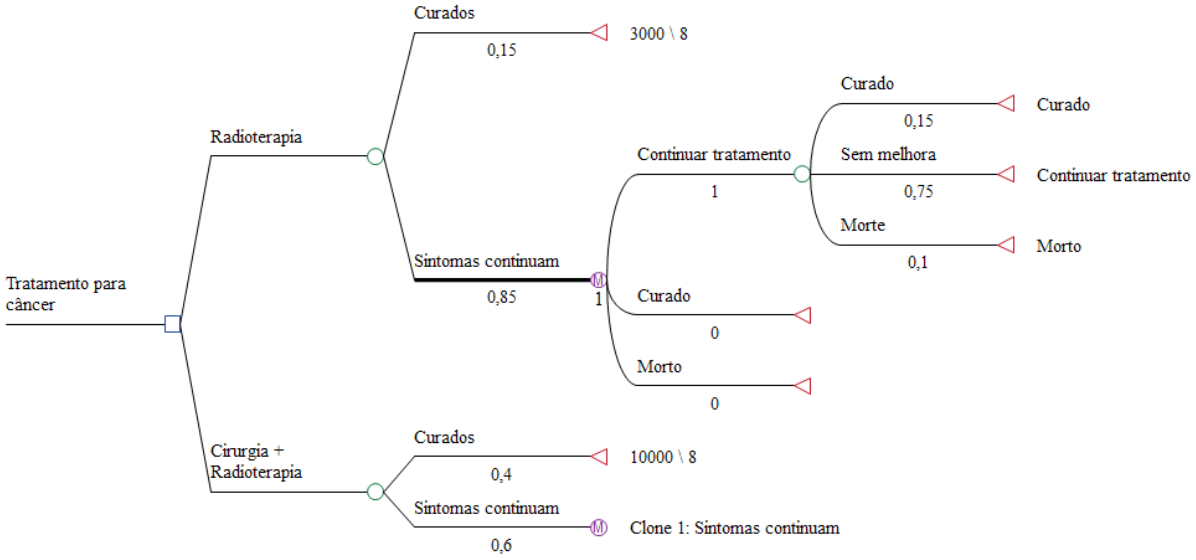

Figura 6 - Representação de um modelo de Markov utilizado em conjunto com uma árvore de decisão.

2.2.1. Construindo um modelo de Markov

O primeiro passo na construção de um modelo de Markov é a definição dos estados que farão parte do modelo, por meio de um diagrama de influência. Os estados devem ser escolhidos de forma a representar condições e acontecimentos que possuam relevância clínica ou econômica. O conjunto formado por todos os estados deve ser finito, e todos os estados devem representar situações mutuamente exclusivas, uma vez que um paciente só pode estar em um estado de cada vez.

O tempo em um modelo de Markov é dividido em partes chamadas ciclos. Todos os ciclos representam um intervalo de tempo de mesmo tamanho, e definem o menor período de tempo que precisa decorrer para que uma mudança possa acontecer. O tamanho de cada ciclo também é definido pelo analista de forma a representar um período de tempo que possua relevância clínica.

A movimentação dos pacientes pelos estados é governada pelas probabilidades de transição. Estas representam a proporção de pacientes que irão permanecer em um mesmo estado, e também a proporção que irá mudar de estado ao final de cada ciclo do modelo.

A Tabela 1 apresenta as probabilidades de transição para o modelo da Figura 5. A linha representa o estado atual do paciente, e a coluna o estado no próximo ciclo. Por exemplo, o valor 0,80 define que 80% dos pacientes que estão no estado "Saudável" irão permanecer nesse mesmo estado ao final de um ciclo. Note que as probabilidades em cada linha da tabela devem somar um, pois isso garante que o número de pacientes que entra no modelo seja igual ao que chega até o final, da mesma forma como em uma árvore de decisão.

Tabela 1 - Matriz de probabilidades de transição para um modelo de Markov.

		Para		
		Saudável	Doente	Morto
De	Saudável	0,80	0,15	0,05
	Doente	0,00	0,70	0,30
	Morto	0,00	0,00	1,00

As probabilidades podem ser constantes durante todo o modelo, ou podem variar conforme o passar do tempo, definindo dois tipos de modelos de Markov. Quando as probabilidades permanecem constantes o modelo é chamado de Cadeia de Markov, e quando as probabilidades variam com o passar do tempo ele é chamado de Processo de Markov.

O modelo é executado a partir de uma configuração inicial que define a quantidade de pacientes em cada estado durante o primeiro ciclo. A partir disso, são utilizadas as probabilidades de transição para determinar o número de pacientes em cada estado para o ciclo seguinte. Este processo é repetido até que todos os pacientes cheguem a um estado do qual não possam mais sair, que é chamado de estado absortivo.

2.2.2. Calculando os resultados

Para obter medidas de custo e de efetividade são necessárias informações individuais para todos os estados. Essas informações refletem o custo de um paciente durante um ciclo e a efetividade associada a cada um dos estados do modelo.

Em todos os ciclos são utilizadas as probabilidades de transição para determinar a quantidade de pacientes em cada um dos estados. Essas quantidades são multiplicadas pelo custo associado a cada estado, gerando o custo total daquele ciclo. Ao final, os custos de todos os ciclos do modelo são somados, gerando o custo total, que se dividido pelo total de pacientes que passaram pelo modelo resulta no custo médio por paciente. O mesmo processo é realizado utilizando alguma medida de efetividade como, por exemplo, a qualidade de vida.

2.2.3. Limitações dos modelos de Markov

Uma dificuldade que aparece quando se trabalha com modelos de Markov diz respeito à sua característica de falta de memória. Todos os pacientes que estão em um mesmo estado são tratados de maneira idêntica, o que nem sempre é verdade, uma vez que os riscos associados a certo paciente podem depender do número de vezes que este já ficou doente, por exemplo. No entanto, essa dificuldade pode ser facilmente contornada (HAWKINS, SCULPHER e EPSTEIN, 2005).

A necessidade de se definir ciclos é considerada como sendo uma das limitações da utilização dos modelos de Markov. Como o ciclo representa o menor espaço de tempo necessário para que uma mudança ocorra no estado de saúde do paciente, sempre irá existir uma perda de informação durante a modelagem, uma vez que não se sabe exatamente quando cada acontecimento ocorre.

Outra limitação dos modelos de Markov é a impossibilidade de incorporar qualquer tipo de dependência entre os pacientes que fazem parte da simulação, ou seja, todos os pacientes precisam ser tratados de maneira independente. Existem situações em que é preciso que os acontecimentos que afetam um paciente influenciem o que acontece com os demais, como quando existe uma capacidade limitada de tratamento, ou quando se deseja trabalhar com doenças infecciosas, por exemplo. Nessas situações é preciso utilizar alguma outra técnica que possa incorporar esta característica, como a simulação de eventos discretos, que será vista a seguir.

2.3. Simulação de Eventos Discretos

Das três técnicas descritas neste trabalho, a Simulação de Eventos Discretos é a que possui maior grau de complexidade, permitindo que sejam criados modelos capazes de descrever adequadamente os mais diversos tipos de problemas de decisão.

Um modelo de eventos discretos possui três componentes que são fundamentais para seu funcionamento: as entidades, os eventos e o tempo.

A entidade representa os elementos que se movem pelos caminhos do modelo e que podem interagir entre si. Em geral, quando se quer avaliar uma tecnologia em saúde essas entidades serão os pacientes, mas também poderiam representar outras pessoas ou objetos, dependendo da necessidade do problema.

Um evento é definido de maneira bastante geral como sendo qualquer acontecimento que afeta os pacientes e que se queira representar no modelo. A ordem de ocorrência desses eventos não precisa ser fixada, podendo ser aleatória, e o modelo considera o exato tempo de ocorrência do evento. Também é possível que esses eventos ocorram mais de uma vez para um mesmo paciente, ou que dois ou mais eventos ocorram simultaneamente. A ordem e as taxas de ocorrência dos eventos podem seguir qualquer distribuição de probabilidade, dependendo do caso em questão, e podem ser alteradas com o passar do tempo.

Os pacientes movem-se pela estrutura do modelo, e são afetados pela ocorrência dos eventos, que podem ocorrer em qualquer instante do tempo. O modelo é capaz de trabalhar com o tempo sem que seja preciso definir intervalos em que os eventos irão ocorrer.

Comparando com um modelo de Markov cujo ciclo tem duração de um ano, se um paciente teve o evento no dia 1 do ciclo e outro paciente teve no dia 359, no modelo de Markov estas duas ocorrências serão tratadas igualmente, enquanto que na simulação de eventos discretos o tempo de cada evento é diferenciado. A Figura 7 faz uma comparação da maneira como o tempo é trabalhado nessas duas técnicas.

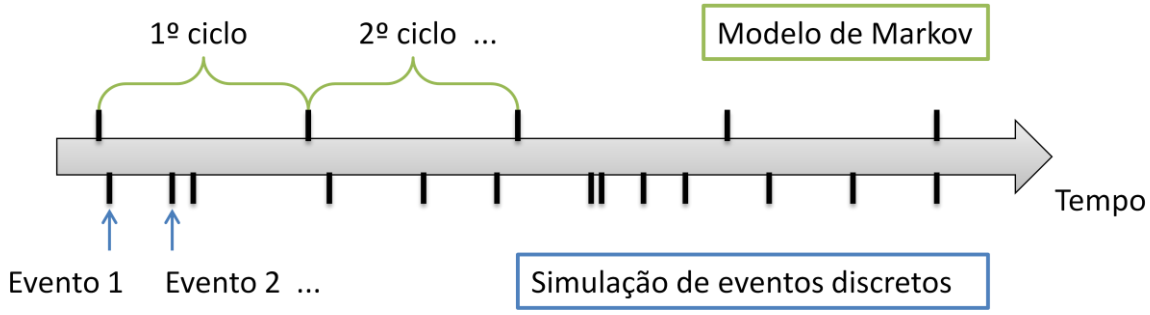

Figura 7 - Representação do tempo em modelos de Markov e de eventos discretos.

Essa característica permite que o tempo seja medido precisamente, ou seja, é possível saber exatamente por quanto tempo um paciente ficou doente, ou quantas horas se passaram desde que ele entrou no hospital, por exemplo.

Outra característica importante dos modelos de eventos discretos são os atributos. Cada paciente que está no modelo pode receber atributos que representam características dos pacientes que são importantes para o modelo, podendo influenciar o caminho percorrido por eles e afetar a ocorrência dos eventos. Podem ser, por exemplo, a idade e o sexo do paciente, qual foi o último tratamento que ele recebeu, ou indicar a presença de fatores de risco, entre outros. Estes atributos podem ser definidos antes do paciente entrar no modelo, mas também podem ser criados e modificados por meio dos eventos que afetam o paciente. Desta forma as probabilidades de ocorrência de eventos podem depender dos atributos do paciente.

Os custos podem ser incluídos no modelo de várias formas. Podem depender dos atributos dos pacientes, ou serem referentes aos custos de determinada intervenção. Dependendo do problema que se quer resolver, pode-se medir o custo total ao longo do tempo, ou medir o custo individual de cada paciente.

O modelo também permite que sejam incluídos recursos que podem ser utilizados pelos pacientes durante o tratamento, como o número de leitos em uma clínica, ou a quantidade disponível de um medicamento. Um paciente pode consumir qualquer quantidade necessária de um ou mais desses recursos por um determinado período de tempo. Isso pode fazer com que certos recursos se tornem indisponíveis para os demais pacientes, podendo levar ao aparecimento de uma fila de espera até que novas unidades desses recursos se tornem disponíveis.

A principal diferença entre a simulação de eventos discretos e as duas outras técnicas de modelagem descritas neste trabalho é a forma como os pacientes são incorporados ao modelo. Cada um deles é tratado de maneira separada em relação aos demais e, portanto, o caminho seguido por eles pode depender de suas características individuais, além de permitir que exista uma interação entre cada um desses pacientes.

Todas essas características fazem com que a simulação de eventos discretos seja capaz de refletir com maior realismo e flexibilidade grande parte das situações nas quais é aplicada a análise de tecnologias em saúde. Porém, para que isso seja possível, é preciso que se tenha uma quantidade maior de informações a respeito do problema em estudo, e que estas informações sejam de melhor qualidade, em comparação com as demais técnicas.

2.3.1. Construindo um modelo de eventos discretos

A principal característica que deve ser levada em conta na hora de construir um modelo para a simulação de eventos discretos é a forma com que o tempo é representado. Os eventos que acontecem durante a simulação ditam como será a passagem do tempo. A partir da definição de quando irá ocorrer cada evento, a simulação avança em saltos entre cada um desses eventos.

As ocorrências dos eventos podem ser definidas a partir de uma regra determinística, como o horário de funcionamento de uma clínica, por exemplo. Também podem depender de um número aleatório gerado a partir de uma distribuição de probabilidade qualquer, ou ainda depender da ocorrência de um evento anterior, como a internação de um paciente após ele sofrer um infarto, por exemplo.

Como os indivíduos no modelo são avaliados separadamente, também é preciso definir de que forma eles serão adicionados ao modelo. Pode ser utilizada uma coorte de pacientes definida no início da simulação, ou então pode ser criada uma regra para determinar a frequência de chegada de novos pacientes. Esta definição depende, evidentemente, da situação que está sendo avaliada.

A maneira como os eventos são incluídos no modelo e a forma como afetam os indivíduos variam conforme o *software* que é utilizado. Em grande parte dos programas criados especificamente para a simulação de eventos discretos o usuário utiliza diversos tipos de peças, cada uma possuindo um tipo diferente de função, para representar o modelo desejado. É através da junção de várias dessas peças que o modelo é construído. Também existe a possibilidade de se utilizar linguagens de programação para criar o modelo, porém essa alternativa requer muito mais conhecimento sobre a teoria que envolve a técnica.

Depois de completa a criação do modelo, a simulação é executada até que certa condição seja satisfeita. Geralmente o modelo é avaliado por um período fixo de tempo, ou até que passe por ele um número pré-determinado de pacientes.

2.3.2. Resultados de uma simulação de eventos discretos

Durante a criação do modelo são definidas quais as medidas que são de interesse para realizar a avaliação. É possível obter praticamente qualquer medida que se tenha interesse, graças à grande flexibilidade que a técnica oferece.

Em uma análise de custo-efetividade é possível obter custos e medidas de efeito para cada um dos pacientes, bem como os valores médios de todos os pacientes que passaram pelo modelo. Em geral, essas medidas podem dizer respeito a cada paciente individualmente, como o custo total gerado por ele durante o modelo, ou o tempo de espera até receber um tratamento, ou ainda se referir a mais de um paciente, como o número de cirurgias realizadas durante um ano, ou a utilização total de um determinado recurso, por exemplo.

2.3.3. Limitações da simulação de eventos discretos.

Uma das desvantagens da técnica é o fato de ela ser mais complexa que as árvores de decisão e os modelos de Markov, sendo necessário alguém com um bom conhecimento de modelagem para criar o modelo. Isso pode fazer com que o economista em saúde precise recorrer a alguém com maior familiaridade com a técnica, possivelmente reduzindo seu controle sobre o modelo. É importante salientar que um bom conhecimento da técnica é necessário não só para criar o modelo, mas também para poder entendê-lo e para interpretar seus resultados.

Outra possível desvantagem diz respeito aos recursos necessários para a criação e execução do modelo, que podem ser maiores com o uso desta técnica em comparação com as outras, aumentando principalmente a quantidade de tempo necessário para o desenvolvimento e execução do modelo.

Por fim, a técnica permite que o problema seja representado com grande nível de detalhes, porém, recomenda-se não tornar o modelo demasiadamente complexo, uma vez que isso aumenta muito a quantidade de informações que o modelo necessita, além de tornar mais difícil o seu desenvolvimento.

3. Escolha do modelo

Muitas das avaliações de tecnologias em saúde que são realizadas utilizam algum tipo de simulação para poder encontrar os resultados necessários ou extrapolar os resultados já existentes, porém pouco se fala sobre a escolha da técnica ou da estrutura empregada para a construção do modelo de simulação. Os resultados obtidos em uma análise deste tipo podem depender muito da técnica utilizada, o que torna essa escolha uma parte fundamental em qualquer avaliação executada.

Diversos autores sugerem que deve ser escolhida a técnica de modelagem mais simples possível, desde que essa seja capaz de atingir os objetivos desejados para o estudo em questão e que possa representar corretamente a estrutura da doença e dos tratamentos avaliados. Outros autores, como BRENNAN, CHICK e DAVIES (2006) propõem uma classificação para diversas técnicas de modelagem, descrevendo detalhadamente cada uma delas, e propondo um passo a passo para a escolha da técnica mais adequada. BARTON, BRYAN e ROBINSON (2004) sugerem que os passos mostrados na Figura 8 sejam seguidos para auxiliar na escolha da técnica.

O primeiro ponto levantado neste fluxograma é se os indivíduos podem ou não serem tratados de maneira independente. Um modelo na qual os indivíduos não podem ser tratados de maneira independente implica que existe alguma forma de interação entre os pacientes dentro do modelo. Isso pode ocorrer, por exemplo, quando se deseja trabalhar com doenças infecciosas, em que a probabilidade de um indivíduo ser contaminado pela doença depende de quantas outras pessoas possuem a doença, ou em situações que tenham uma quantidade escassa de recursos, e a escolha sobre o tratamento que é dado a um paciente influencia o que pode ser dado para outro.

Quando a situação não envolve nenhum tipo de interação entre os pacientes, a escolha da técnica é feita principalmente entre árvores de decisão e modelos de Markov, que trabalham com os pacientes agrupados em coortes. Porém, em alguns casos, pode ser necessário utilizar um modelo de amostragem de indivíduos, também conhecido como simulação de Monte Carlo de primeira ordem ou microssimulação, que utiliza uma árvore de decisão ou um modelo de Markov para simular cada paciente individualmente, permitindo que seja incluído um tipo de memória do que acontece com cada paciente.

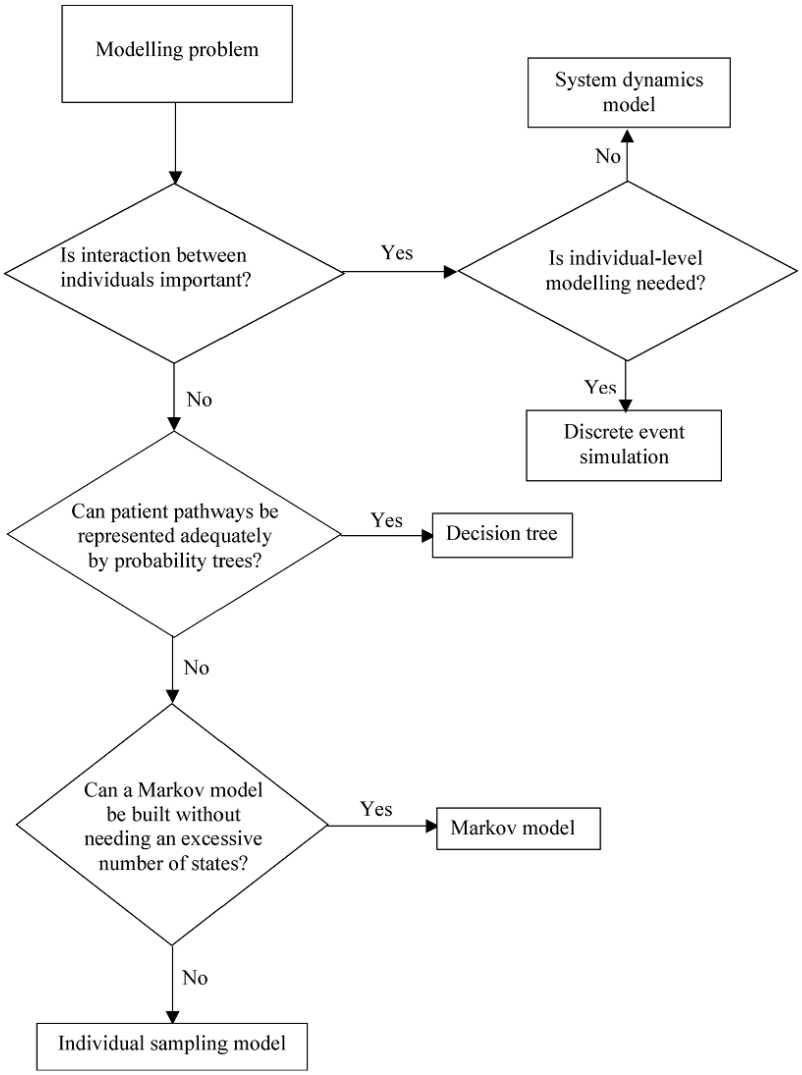

Figura 8 - Fluxograma proposto por BARTON, BRYAN e ROBINSON (2004) para a escolha da técnica de modelagem.

A árvore de decisão mostra de maneira explícita todos os possíveis caminhos que podem ser percorridos pelos pacientes, com as probabilidades associadas a cada um deles e as medidas de custo e de efetividade para cada possível desfecho. Essas medidas são ponderadas pelas probabilidades para gerar o custo e efetividade esperada para cada uma das opções. Normalmente as árvores de decisão possuem um nodo de decisão em seu início, e a partir dele é gerado um conjunto de árvores de probabilidade, sendo uma para cada programa de ação.

O uso da árvore de decisão é mais indicado quando se deseja avaliar um curto período de tempo. Qualquer problema que trate os pacientes como sendo independentes entre si poderia ser representado por uma árvore. Porém, conforme o tamanho desta aumenta devido à ocorrência de muitos eventos ou de muitas ocorrências de um único evento, o problema começa a se tornar cada vez mais difícil de ser manuseado, tornando necessário o uso de outra técnica que resulte em um modelo mais simples.

Por sua vez, os modelos de Markov possuem a característica de conseguirem trabalhar facilmente com eventos recorrentes, levando em conta explicitamente a passagem do tempo, que é dividido em ciclos de igual comprimento e representam o menor intervalo de tempo considerado relevante para o problema em estudo. Em qualquer instante do tempo cada paciente pode estar em um dos possíveis estados, dentre um conjunto finito destes. Para cada par de estados existe uma probabilidade de transição associada, que é a probabilidade condicional de o paciente estar no segundo estado no final do ciclo, dado que estava no primeiro estado no início deste ciclo.

Em cada ciclo do modelo de Markov é calculada a proporção de pacientes que está em cada um dos estados, e os custos são acumulados de acordo com o número de pacientes. Diferentes estratégias de ação podem ser testadas alterando o custo e as probabilidades associadas a cada estado.

As principais limitações dos modelos de Markov são o agrupamento de pacientes e a divisão do tempo em ciclos, mas estas podem ser superadas com o uso de simulações de Monte Carlo para criar um modelo de amostragem de indivíduos, em que um único indivíduo com características próprias é simulado em cada execução do modelo.

Quando a interação entre os indivíduos é importante para o modelo, técnicas como a simulação de eventos discretos ou os sistemas dinâmicos se fazem necessárias.

A simulação de eventos discretos permite que seja representado completamente o histórico de cada paciente, em termo de suas características basais e dos acontecimentos que ocorreram durante a simulação, bem como as possíveis interações que possam surgir entre os pacientes. O preço a ser pago para que se tenha tal nível de detalhamento no modelo é a necessidade de serem utilizados programas especializados ou o uso de programação para poder construir e executar esses modelos. O tempo de execução também costuma ser muito maior em comparação com as outras técnicas, devido à necessidade de calcular cada indivíduo separadamente e todos os eventos que podem acontecer para cada um deles.

Os modelos de sistemas dinâmicos trabalham com os pacientes agregados e permitem certas formas de interação entre eles, além de serem bastante rápidos de serem executados, porém não conseguem incorporar completamente o histórico de cada paciente. Esta técnica não será abordada neste trabalho, e uma melhor descrição pode ser vista nos trabalhos de PIDD (2004) e BRAILSFORD (2008).

Quando a interação entre indivíduos é bastante importante e precisa-se trabalhar com cada indivíduo separadamente, um modelo de eventos discretos é a única alternativa para a situação. Mesmo com os computadores atuais a execução de um modelo deste tipo leva um tempo considerável. O problema ainda piora com a necessidade de rodar o mesmo modelo diversas vezes para estimar adequadamente as médias populacionais do conjunto de parâmetros testados na presença da variabilidade. Porém, estes problemas não justificam a troca da simulação de eventos discretos por técnicas mais simples se estas omitem características que são importantes para o problema em estudo.

Existem ocasiões em que se precisa de uma resposta rápida para um problema em decisão e então um modelo simples é criado sem que sejam testadas as suposições envolvidas. Por outro lado, algumas decisões precisam de modelos que possam ser adaptados com o passar do tempo para avaliar diferentes populações e também para poder incluir novas evidências em relação a fatores de risco ou a novos tratamentos. Nessas situações o analista deve considerar a utilização de modelos individuais, em particular o modelo de eventos discretos, por ser uma técnica bastante flexível (BRENNAN, CHICK e DAVIES, 2006).

A simplicidade na construção dos modelos é sempre uma característica desejável, principalmente no que diz respeito ao tamanho e a complexidade empregada em sua construção, uma vez que modelos mais simples são mais fáceis de serem entendidos do que modelos mais complexos, e por isso são mais fáceis de serem validados.

Situações mais complexas precisam de modelos que sejam capazes de lidar com tal complexidade. Existem técnicas para estas situações e essas precisam ser adotadas mais amplamente pela comunidade de economia em saúde. Pode ser preciso mais habilidade para construir um modelo utilizando tais técnicas, mas é preciso adquirir tal habilidade para não correr o risco de produzir recomendações inapropriadas como resultado da utilização de um modelo inadequado para a situação (BARTON, BRYAN e ROBINSON, 2004).

4. Exemplo

Neste capítulo será abordada a construção de um modelo baseado no trabalho feito por JAHN *et al.* (2010). Este exemplo analisa uma situação em que é avaliado o tratamento de pacientes com doença arterial coronariana, e tem como objetivo principal apresentar um passo a passo que demonstre como construir um modelo de eventos discretos utilizando o *software* MATLAB, além de demonstrar as diferenças que podem ocorrer nos resultados das simulações em decorrência da técnica utilizada para realizar a modelagem. Em nenhum momento este trabalho terá a preocupação de reproduzir ou comparar de qualquer maneira os resultados obtidos com aqueles que foram encontrados pelos autores do exemplo.

A doença arterial coronariana é caracterizada pelo estreitamento ou obstrução das artérias coronárias. Os principais tratamentos para essa doença são a angioplastia coronária e a cirurgia de revascularização do miocárdio. A angioplastia envolve a aplicação de um cateter com um balão em sua ponta que, quando inflado, desobstrui a artéria estreitada. Para diminuir o risco de reestenose (reobstrução), uma endoprótese metálica (*stent*) é utilizada.

Neste exemplo, as *stents* convencionais são comparadas com um novo tipo de *stent* que é revestida com fármacos. Com o uso das *stents* metálicas convencionais, o risco de reestenose ainda pode ser bastante elevado, porém, as novas *stents* revestidas com fármacos têm mostrado resultados promissores, levando a um menor número de intervenções repetidas.

Os pacientes incluídos na análise são classificados em quatro subgrupos, dependendo tanto da presença ou não de diabetes, quanto do tipo de lesão e do tamanho da artéria afetada. A Tabela 2 apresenta a descrição dos quatro subgrupos criados a partir desses dois fatores. Esta classificação é realizada uma vez que o risco de reestenose pode variar significativamente conforme esses fatores de risco.

Tabela 2 - Descrição dos quatro subgrupos avaliados.

Subgrupo	Diabetes	Tipo de lesão e tamanho da artéria
1	Sem diabetes	Lesão longa ou artéria estreita
2	Sem diabetes	Lesão curta e artéria larga
3	Com diabetes	Lesão longa ou artéria estreita
4	Com diabetes	Lesão curta e artéria larga

São avaliados quatro diferentes estratégias de tratamento, sendo que em cada uma delas é escolhido um tipo diferente de *stent* para ser aplicado ao paciente, de acordo com o subgrupo em que este foi classificado. Por exemplo, os pacientes que não possuem diabetes e que tiveram lesões longas ou vasos estreitos afetados irão receber as *stents* convencionais em duas das estratégias, e receberão as novas *stents* nas outras duas estratégias avaliadas.

Também é avaliado o número máximo de angioplastias que podem ser realizadas por dia. Esse número é definido pela quantidade de instalações e de pessoal disponível. Se a demanda exceder a capacidade, as próximas intervenções são adiadas, levando ao aparecimento de filas de espera e à necessidade de tratamento adicional para esses pacientes.

O modelo a ser construído deve acomodar a seguinte situação: pacientes necessitando de uma angioplastia chegam a certa unidade de saúde a uma taxa de 40 pacientes por dia. Se houver capacidade o tratamento é realizado na data de chegada, caso contrário os pacientes são colocados em uma fila de espera.

A Figura 9, retirada do artigo de JAHN *et al.* (2010), ilustra a situação avaliada.

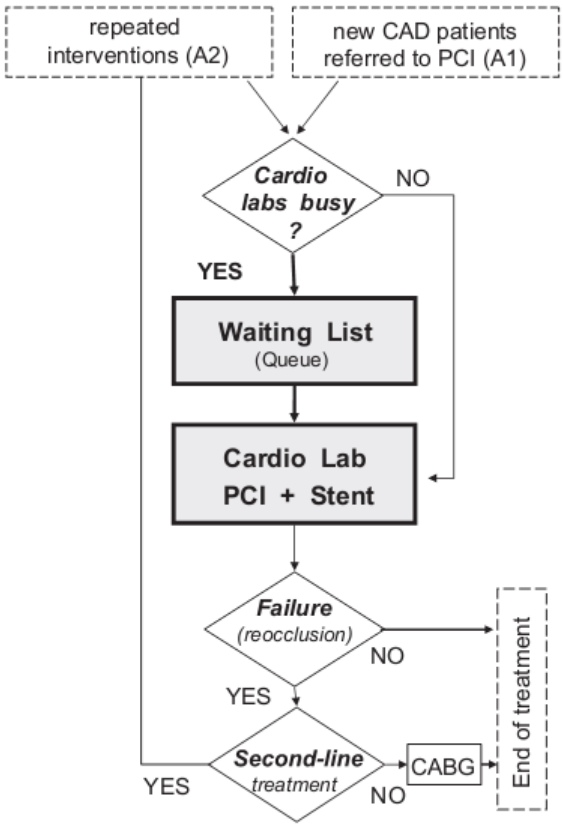

Figura 9 - Fluxo de pacientes para o tratamento da doença arterial coronariana (CAD) incluindo a fila de espera. CABG, cirurgia de revascularização do miocárdio; PCI, angioplastia coronária.

Assume-se que durante um ano após a intervenção ainda exista a possibilidade de recorrência da doença. Se isso acontecer o paciente é encaminhado para receber uma nova intervenção. Caso contrário, considera-se que o paciente está curado e encerra-se o seu acompanhamento. Se mesmo após a segunda intervenção a doença se manifestar novamente em um período de um ano, o paciente é encaminhado para receber uma cirurgia de revascularização do miocárdio.

Uma quantidade maior de reestenoses na população estudada leva a uma quantidade maior de intervenções. Logo, se as novas tecnologias reduzirem o número de intervenções repetidas, isso pode acarretar uma melhora nas filas de espera, nos custos de tratamento e na qualidade de vida dos pacientes.

A necessidade de incorporar uma fila de espera para os pacientes que irão receber o atendimento caracteriza a necessidade de se utilizar um modelo de eventos discretos. Para fins de comparação, será criada uma estrutura de árvore de decisões sem a inclusão do limite diário de atendimentos, que será comparada com a mesma situação recriada com um modelo de eventos discretos. O modelo de eventos discretos também será usado para avaliar as diferenças observadas na custo-efetividade quando for considerado um número máximo de atendimentos por dia.

4.1. Parâmetros do Modelo

As estimativas para os parâmetros utilizados na construção deste exemplo foram retirados do artigo escrito por JAHN *et al.* (2010).

O modelo de eventos discretos será avaliado considerando um período de sete anos, com novos pacientes chegando diariamente para receber o tratamento. Supõe-se que o número de pacientes que chegam a cada dia pode ser representado por uma distribuição de Poisson com taxa $\lambda = 14600/365$. Essa taxa expressa o número médio de pacientes que chegam por dia para receber o tratamento, ou seja, chegam 40 novos pacientes por dia, em média (14600 por ano). Isso será incorporado ao modelo utilizando uma distribuição Exponencial com média $1/\lambda = 365/14600$ para determinar o tempo entre a chegada de cada paciente.

A Tabela 3 apresenta a proporção de cada um dos subgrupos na população e o tipo de *stent* recebido pelos pacientes de cada subgrupo para as quatro estratégias avaliadas. A letra B significa *bare-metal stents*, que são as *stents* convencionais, e a letra D significa *drug-eluting stents*, que são as *stents* revestidas com fármacos.

Tabela 3 - Proporção de indivíduos em cada subgrupo e estratégias de tratamento.

	Proporção	Estratégia			
		1	2	3	4
Subgrupo 1	0,38	B	B	D	D
Subgrupo 2	0,40	B	B	B	B
Subgrupo 3	0,12	B	D	D	D
Subgrupo 4	0,10	B	B	B	D

A estratégia 1 (BBBB) considera que todos os pacientes recebem as *stents* convencionais. As outras estratégias são definidas de acordo com as taxas de recorrência de cada subgrupo com a utilização dessas *stents*. Na estratégia 2 (BBDB) os pacientes do subgrupo 3 recebem as novas *stents*, pois são os que possuem o maior risco de recorrência com as *stents* convencionais. Na estratégia 3 (DBDB) os pacientes dos subgrupos 1 e 3 são os que recebem as novas *stents*, e na estratégia 4 (DBDD) o mesmo acontece para os pacientes dos subgrupos 1, 3 e 4.

O subgrupo 2 recebe as *stents* convencionais em todas as estratégias consideradas, pois a taxa de recorrência estimada com a utilização das novas *stents* é maior do que aquela estimada para as *stents* convencionais. A Tabela 4 mostra as taxas anuais de recorrência, os custos de cada intervenção, e os valores de qualidade de vida associados.

Tabela 4 - Parâmetros utilizados no modelo.

	Stents metálicas convencionais	Stents revestidas com fármacos	Cirurgia de revascularização do miocárdio
Taxa anual de revascularização			
Subgrupo 1	0,095	0,054	-
Subgrupo 2	0,051	0,054	-
Subgrupo 3	0,143	0,069	-
Subgrupo 4	0,055	0,051	-
Utilização de recursos			
Custo da intervenção	€ 1.953	€ 2.648	€ 6.560
Custo da internação	€ 2.808	€ 2.808	€ 16.180
Custo de espera (por dia)	€ 15	€ 15	-
Valores de qualidade de vida			
Baseline	0,69	0,69	0,68
1 mês	0,84	0,84	0,78
6 meses	0,86	0,86	0,86
12 meses	0,86	0,86	0,87

O interesse dessa avaliação é medir a qualidade de vida média dos pacientes do instante que estes são incluídos no modelo até eles serem completamente tratados, isto é, quando ficarem um ano sem apresentar a recorrência da doença ou após realizarem a cirurgia.

A qualidade de vida média ao longo do tempo é obtida através do cálculo da área embaixo da curva formada pelos pontos que representam a qualidade de vida medida em determinados instantes (WILLAN e BRIGGS, 2006).

A Figura 10 mostra a curva da qualidade de vida estimada para um paciente que recebeu a aplicação de *stents*, no período de um ano. Esse paciente precisou receber somente a primeira intervenção para ser curado. Antes de receber a intervenção, enquanto ainda sofria com a doença, a qualidade de vida desse paciente era de $0{,}69$. Um mês depois de receber o tratamento a sua qualidade de vida já é igual a $0{,}84$, aumentando até $0{,}86$ após seis meses, e permanecendo nesse valor até completar um ano.

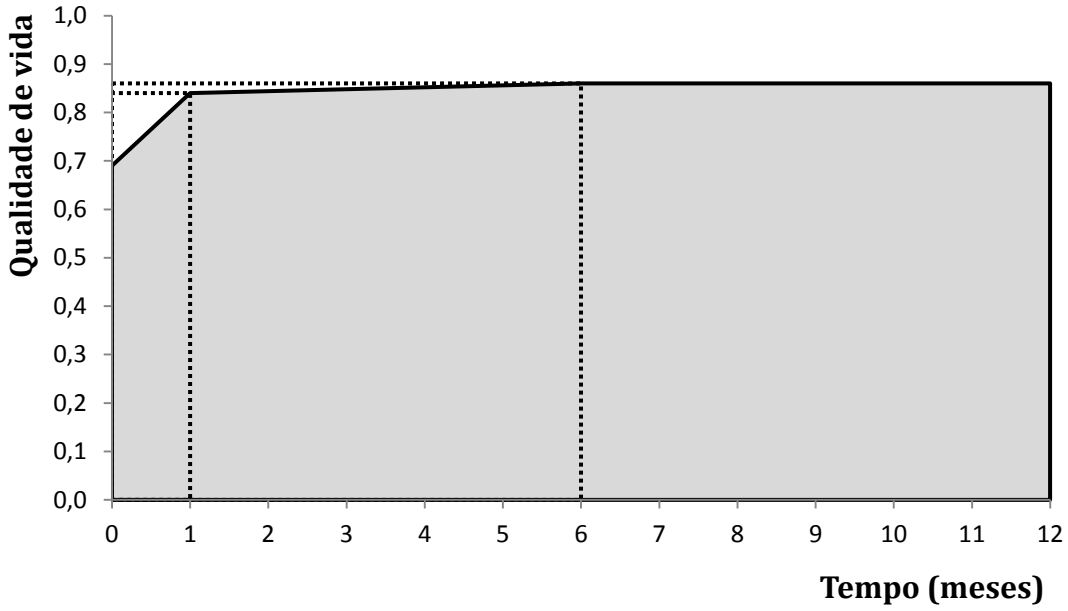

Figura 10 - Curva estimada para a qualidade de vida dos pacientes, em um ano.

Para calcular a área é preciso encontrar o valor médio em cada segmento de reta e multiplicar este valor pelo comprimento do segmento. Por exemplo, para o segmento entre 1 e 2 meses, o valor é a média entre os valores $0{,}69$ e $0{,}84$, que é igual a $0{,}765$. Multiplicando esse valor pelo comprimento da reta, que é igual a 1, tem-se que a qualidade de vida nesse período foi igual a $0{,}765$ mês. Após calcular esse valor para os outros 2 segmentos de reta, são obtidos os valores $4{,}25$ e $5{,}16$, respectivamente. Somando os três valores, e dividindo pelo comprimento total do intervalo, que é de doze meses, obtém-se $10{,}175/12 = 0{,}8479$, representando a qualidade de vida média desse paciente nesse período de um ano.

4.2. Modelo de Árvore de Decisão

O exemplo que é demonstrado aqui pode ser criado através de uma árvore de decisão quando o limite na capacidade diária de atendimentos é retirado. Fazendo isso o caminho percorrido por cada paciente passa a ser independente dos demais, uma vez que todos podem ser atendidos imediatamente. Neste caso, é possível calcular os custos e a qualidade média para cada paciente, bem como a probabilidade associada ao número de intervenções que precisam ser realizadas para que um paciente seja curado.

As probabilidades de recorrência podem ser facilmente obtidas através da fórmula para transformar taxas em probabilidades: $p = 1 - \exp(-\lambda * \mathrm{t})$, onde λ representa a taxa e t o tempo. Para ter a probabilidade de um paciente sofrer uma recorrência no período de um ano utilizam-se as taxas anuais de recorrência e $t = 1$. Assim são obtidas as probabilidades de recorrência com cada tipo de *stent* para cada subgrupo, que são mostradas na Tabela 5.

Tabela 5 - Probabilidade de recorrência em um ano, para cada subgrupo.

Subgrupo	Proporção	Probabilidade de recorrência	
		Stents Convencionais	*Stents* com Fármacos
1	0,38	0,05257	0,09063
2	0,40	0,05257	0,04972
3	0,12	0,06667	0,13325
4	0,10	0,04972	0,05351

Os custos utilizados são a soma dos custos de intervenção e de internação. Além disso, o custo associado aos pacientes que fazem a cirurgia também inclui o custo das duas intervenções anteriores.

Para obter a qualidade de vida média supõe-se que quando um paciente sofre uma recorrência, esta irá ocorrer na metade do período de um ano. Os pacientes que precisarem apenas da primeira intervenção obtém uma qualidade de vida média igual a $0{,}8479$, que foi calculado anteriormente, referente ao período de um ano após a aplicação de um *stent*.

Por sua vez, os pacientes que precisarem de uma segunda intervenção terão permanecido meio ano se recuperando da ultima intervenção, em média, até que a doença se manifeste novamente. Neste meio ano os pacientes terão vivido $0{,}4179$ ano, ajustado pela qualidade de vida, que pode ser obtido pelo mesmo cálculo descrito anteriormente. Depois de receberem a segunda aplicação de *stent* eles irão permanecer mais um ano se recuperando, e a qualidade de vida média desses pacientes será de $(0{,}8479 + 0{,}4179)/1{,}5 = 0{,}8439$.

Os pacientes que não forem curados com as duas intervenções e precisarem receber a cirurgia terão sofrido duas recorrências no período de um ano, resultando em uma qualidade de vida média de $(0{,}4179 + 0{,}4179)/1 = 0{,}8358$. A Tabela 6 mostra os valores médios de custo e de qualidade de vida calculados para os três desfechos.

Tabela 6 - Medidas associadas aos três possíveis desfechos.

Tipo de *stent*	Uma intervenção		Duas intervenções		Cirurgia	
	Custo (€)	Qualidade	Custo (€)	Qualidade	Custo (€)	Qualidade
Convencionais	4761	0,847917	9522	0,843889	32262	0,835833
Com fármacos	5456	0,847917	10912	0,843889	33652	0,835833

Com esses valores é possível criar a árvore de decisão. A Figura 11 mostra o galho da árvore que representa o Subgrupo 1 com a estratégia BBBB, juntamente com os custos e qualidade médios já calculados. A estrutura dos demais galhos da árvore é idêntica a esse, mudando apenas as probabilidades associadas aos nós de chance e os custos associados às intervenções com cada tipo de *stent*.

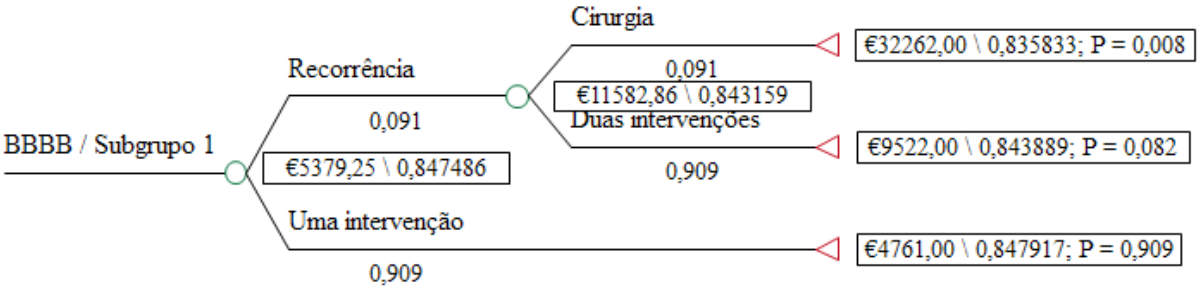

Figura 11 - Representação de parte do modelo de árvore de decisões.

Os valores obtidos através dessa árvore de decisões para o custo médio e a qualidade de vida média em cada uma das quatro estratégias são mostrados na Tabela 7. Por exemplo, com a estratégia BBBB, o custo médio para tratar cada paciente é de 5269,67 e a qualidade de vida média desses pacientes durante o período do tratamento é de 0,8475594. Esses valores serão comparados posteriormente com os obtidos através do modelo de eventos discretos.

Tabela 7 - Medidas finais de custo e efetividade para cada estratégia.

Estratégia	Custo (€)	Qualidade
BBBB	5269,67	0,8475594
BBDB	5284,28	0,8476044
DBDB	5446,32	0,8476793
DBDD	5516,58	0,8476812

4.3. Modelo de Eventos Discretos

Para a criação de um modelo de eventos discretos deve-se escolher primeiramente qual será o *software* utilizado neste processo. Existem diversas opções disponíveis para escolha, dependendo das necessidades e das preferências do analista. As opções vão desde linguagens de programação, que oferecem grande flexibilidade ao custo de serem complexas de se trabalhar, até programas específicos para a simulação de eventos discretos, que são fáceis de serem utilizados, mas podem não serem capazes de tudo o que seria possível de se fazer via programação.

Para a maior parte dos problemas de decisão, os programas específicos são a melhor opção. A escolha do programa que será utilizado depende principalmente das capacidades que cada um deles oferece, devendo ser estudadas de acordo com as necessidades do problema.

Um dos *softwares* capazes de trabalhar com a simulação de eventos discretos é o MATLAB®, desenvolvido pela MathWorks. O MATLAB é utilizado nas mais diversas áreas do conhecimento, possui inúmeras ferramentas para o desenvolvimento de modelos de simulação e será o software utilizado neste trabalho.

Um dos pacotes de ferramentas disponíveis, chamado SimEvents®, é voltado especificamente para o desenvolvimento de modelos de eventos discretos e pode ser utilizado em conjunto com as ferramentas disponíveis em outros pacotes. Isso faz com que o MATLAB seja uma ferramenta bastante flexível para a criação de modelos, sem que se torne demasiadamente complicado de se utilizar.

4.3.1. Configurando um compilador

Antes de começar a trabalhar com o modelo de simulação, é preciso verificar se o MATLAB está configurado corretamente para utilizar as ferramentas de geração de código. Essas ferramentas se tornam disponíveis por meio da utilização de um compilador para a linguagem de programação C/C++.

A versão de 32 bits do MATLAB 2011 fornece um desses compiladores, sendo necessário apenas que esse seja configurado corretamente. A versão de 64 bits, por sua vez, não fornece um em sua instalação e, portanto, é preciso que um compilador compatível seja instalado para que essas ferramentas se tornem disponíveis. A versão do MATLAB pode ser verificada através da opção *Help → About MATLAB.*

Para a versão de 32 bits do MATLAB é preciso somente executar o comando “mex –setup” na janela de comandos. Esse comando dará a opção de buscar automaticamente os compiladores instalados. Respondendo “y” aparecerá uma lista com os compiladores encontrados, e então basta responder “1” e “y” novamente para concluir a configuração.

Para a versão de 64 bits do MATLAB a MathWorks recomenda que seja utilizado o Microsoft Visual C++ 2010 Express, juntamente com o Microsoft Windows SDK 7.1, que são disponibilizados gratuitamente para download através do site da Microsoft. As instruções de instalação para esses dois programas podem ser encontradas no site da Mathworks (MATHWORKS, 2011a). Pode ser que o MATLAB não seja capaz de encontrar o local onde o Microsoft Windows SDK 7.1 foi instalado mesmo após a correta instalação do programa, fazendo com que seja necessária a criação de uma variável de ambiente chamada “MSSdk” contendo o local onde ele foi instalado (MATHWORKS, 2011b). Após certificar-se de que um compilador compatível está disponível no computador, executa-se o comando “mex –setup” na janela de comandos do MATLAB para localizar o compilador e fazer as configurações necessárias, da mesma forma que na versão de 32 bits.

É possível verificar se o compilador foi configurado corretamente digitando o comando “mex.getCompilerConfigurations()” na janela de comandos do MATLAB.

4.3.2. Iniciando um novo modelo

Para iniciar a construção de um novo modelo escolhe-se a opção *File* → *New* → *Model* na janela principal do MATLAB. Essa opção abrirá uma janela contendo um modelo em branco igual à que é mostrada na Figura 12.

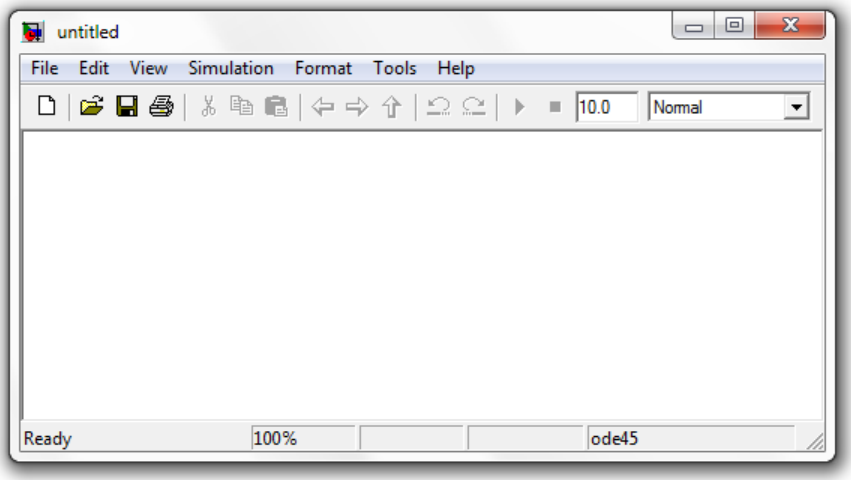

Figura 12 - Janela inicial de um modelo em branco.

Antes de iniciar a construção do modelo é preciso alterar uma configuração necessária para executar a simulação de eventos discretos. Selecionando a opção *Simulation* → *Configuration Parameters* será aberta a janela de configurações da simulação, onde se deve alterar a opção *Solver* de *ode45 (Dormand-Prince)* para *discrete (no continuous states)*. Essa opção faz com que o programa passe a trabalhar com o tempo em intervalos discretos, calculando o instante de tempo em que os eventos ocorrem.

Nessa janela de opções também é possível alterar o tempo de duração da simulação, através das opções *Start time* e *Stop time*. A opção *Stop Time* também pode ser alterada através da barra de ferramentas na janela do modelo. Por padrão, a duração é definida como 10.0, mas pode ser alterada para qualquer valor maior que zero, ou mesmo uma expressão matemática simples. Neste exemplo a duração será definida como $7 * 365$, representando 7 anos. O significado desse número depende completamente da forma como o tempo é tratado pelo modelo.

Para criar o modelo são utilizados diversos tipos de blocos com diferentes funções. É através da junção desses blocos que o caminho percorrido pelos pacientes é representado. Selecionando a opção *View* → *Library Browser* é aberta uma nova janela, mostrada na Figura 13, onde são encontrados todos os blocos que podem ser utilizados na criação dos modelos.

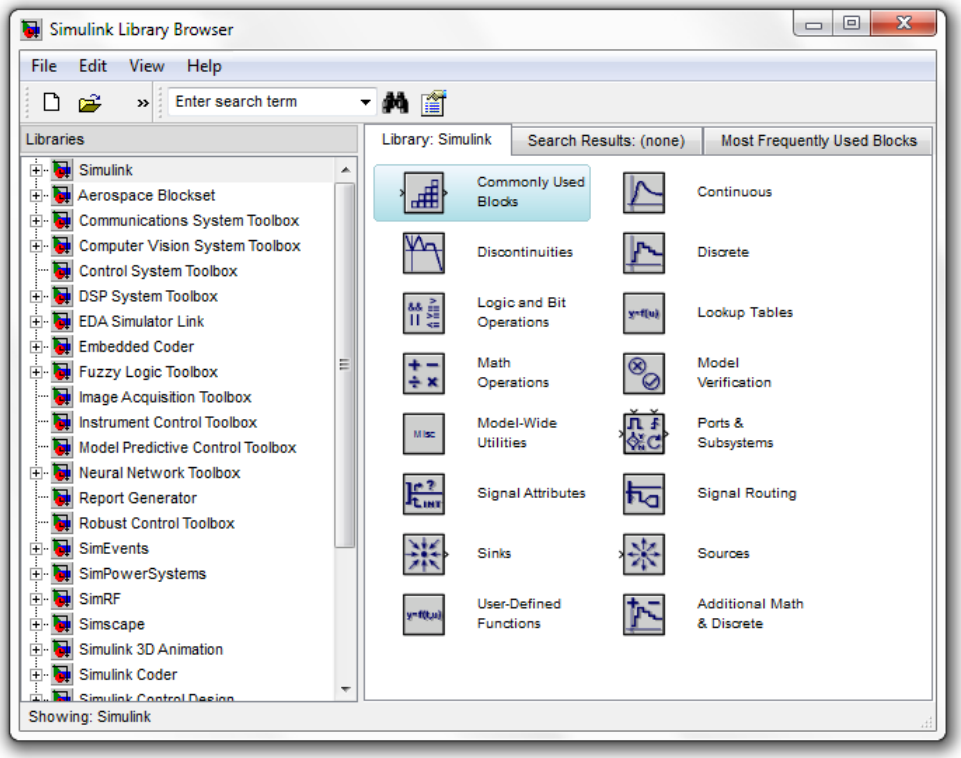

Figura 13 - Biblioteca de blocos para simulação.

Neste exemplo, utilizaremos os blocos de duas das bibliotecas disponíveis. Os blocos básicos, como os que executam operações matemáticas ou lógicas, podem ser encontrados na biblioteca *Simulink*. Já os blocos utilizados especificamente para a simulação de eventos discretos estão na biblioteca *SimEvents*.

4.3.3. Incluindo novos pacientes

Os blocos contidos em uma biblioteca são organizados de acordo com a função que executam. Na biblioteca *SimEvents* existe um grupo chamado G*enerators* que contém dois tipos de geradores de entidades, que podem ser encontrados clicando duas vezes em *Entity Generators*. Um deles depende da ocorrência de algum outro evento para gerar uma nova unidade, enquanto o outro gera novas unidades ao decorrer do tempo.

Neste modelo será incluído um gerador baseado no tempo. Isso pode ser feito clicando e arrastando o bloco da biblioteca até o modelo, ou clicando com o botão direito do mouse em cima do bloco e escolhendo a opção *Add to "Nome do modelo"*.

Pode-se clicar no nome de qualquer bloco para alterá-lo de modo que reflita melhor o que o bloco representa, como em uma caixa de texto. Na Figura 14 o gerador de entidades foi incluído ao modelo e seu nome foi alterado para "Entrada".

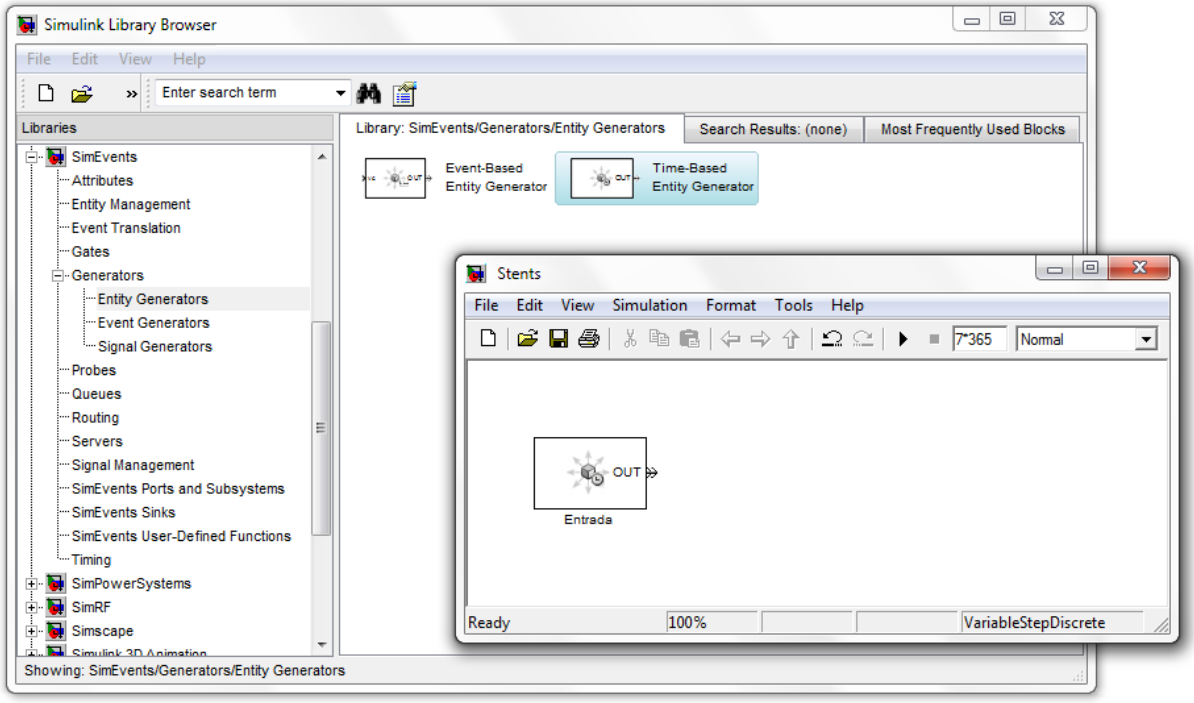

Figura 14 - Janela do modelo após a inclusão do bloco gerador de entidades.

Clicando duas vezes em um bloco temos acesso às suas configurações. O gerador que incluímos permite que seja alterada a taxa que controla o tempo entre a chegada de cada um dos pacientes. As opções disponíveis para controlar o tempo são *Intergeneration time from dialog e Intergeneration time from port t*. A primeira delas especifica que o tempo de chegada de cada paciente será controlado pelo próprio bloco gerador de entidades, utilizando um intervalo constante, uma distribuição aleatória uniforme, ou uma distribuição aleatória exponencial. A segunda permite que as chegadas sejam controladas por outro bloco. Neste exemplo, será utilizado que o tempo entre as chegadas segue distribuição exponencial com média $365/14600$, como foi visto nos parâmetros do modelo. A Figura 15 mostra as configurações utilizadas neste exemplo.

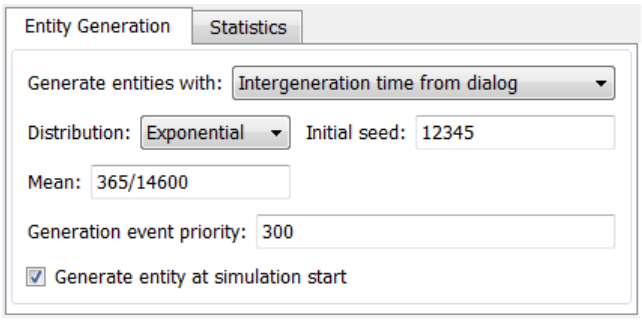

Figura 15 - Configurações do gerador de entidades.

Uma opção importante em todos os geradores aleatórios é a semente, que é utilizada pelo programa para determinar quais serão os próximos números gerados. Todos os geradores incluídos possuem por padrão a semente 12345, mas recomenda-se que esse valor seja alterado em todos os geradores, de modo que cada um dos geradores do modelo possua uma semente diferente dos demais.

Enquanto o modelo estiver sendo construído e testado pode ser útil deixar a semente fixa, para que os resultados possam ser reproduzidos em todas as execuções, facilitando o processo de teste do modelo. Quando o modelo estiver completo e for necessário executá-lo várias vezes, surge a necessidade de se alterar a semente em cada nova execução para representar a variabilidade natural das observações. Uma maneira de fazer isso automaticamente é usar uma expressão para definir uma nova semente em cada execução. Uma das expressões que podem ser utilizadas é $mod(ceil(cputime * k), 2\text{^}32)$, onde k representa uma constante qualquer e que deve ser diferente em cada bloco gerador de números aleatórios. Esta expressão utiliza o tempo total de execução do MATLAB para criar a semente, truncando o valor com a função *ceil* e utilizando a função *mod* para garantir que o valor seja menor que 2^{32}, o máximo permitido.

4.3.4. Atribuindo características

Os pacientes que entram no modelo são classificados de acordo com um dos quatro subgrupos definidos. Neste exemplo cada paciente será classificado aleatoriamente em um dos quatro subgrupos através do resultado obtido de um bloco gerador de números aleatórios, que é chamado *Event-Based Random Number* e que pode ser encontrado na biblioteca *SimEvents Generators - Signal Generators*.

Este bloco gera valores aleatórios provenientes de diversas distribuições de probabilidade. Nas configurações do bloco, escolhe-se a distribuição *Arbitrary discrete* e digitam-se os possíveis subgrupos e as probabilidades de pertencer a cada subgrupo, assim como mostra a Figura 16.

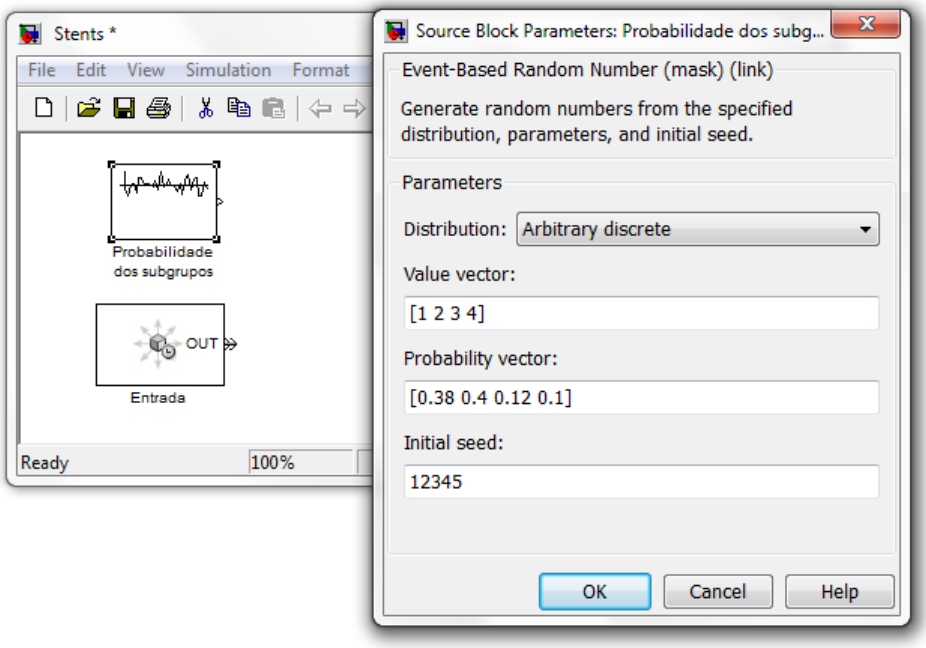

Figura 16 - Gerador de números aleatórios que define o subgrupo.

Agora, é preciso atribuir o subgrupo aos pacientes que entrarem no modelo. Isso é feito através do bloco *Set Attribute*, que cria ou altera os atributos de uma entidade. Esse bloco é encontrado na biblioteca *SimEvents Attributes* e também deve ser incluído ao modelo. Acessando as configurações do bloco especifica-se que o nome do atributo é “Subgrupo”, e é necessário especificar um valor para representá-lo. Esse valor pode ser fixo ou pode depender de outra parte do modelo. A opção *Value From Dialog* atribui o valor que for digitado na caixa *Value* a todos os pacientes que passarem pelo bloco, enquanto que a opção *Value From Signal Port* atribui um valor recebido de outro bloco, podendo variar para cada entidade.

Como o valor do subgrupo de cada paciente será gerado pelo bloco gerador de números aleatórios, seleciona-se a opção *Value From Signal Port*, como mostra a Figura 17.

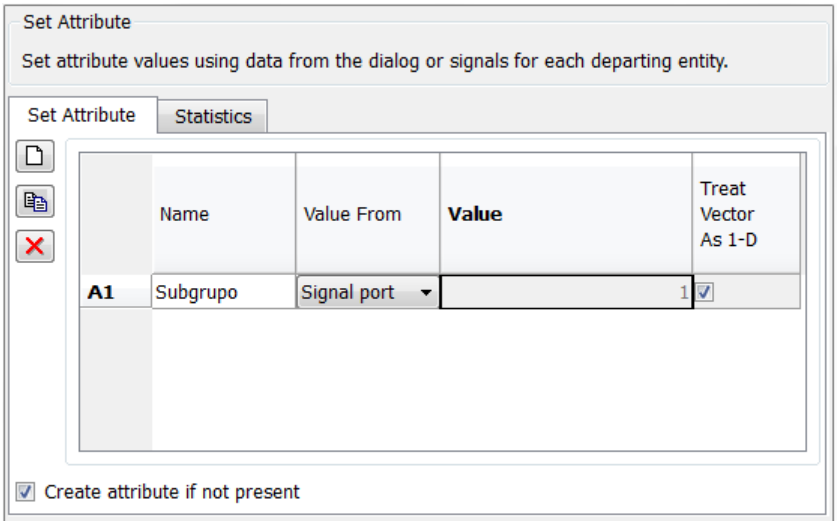

Figura 17 - Propriedades do bloco para salvar atributos.

Essa opção fará com que um novo símbolo representando uma entrada chamada *A1* apareça no bloco *Set Attribute*. Existem dois tipos de entradas e saídas nos blocos que iremos utilizar, dependendo do que está sendo transferido de um bloco para outro.

Em um modelo de eventos discretos podemos ter as entidades, que representam os pacientes que estão no modelo, ou os sinais, que são qualquer valor numérico transferido entre os blocos. O caminho por onde os pacientes passam é representado através de setas duplas, enquanto o caminho dos sinais é representado por uma seta simples.

Para criar uma ligação entre dois blocos, clica-se na saída de um bloco e arrasta-se até a entrada de outro. Vamos fazer isso ligando a saída *OUT* do gerador de entidades até a entrada *IN* do bloco que identifica o subgrupo, e a saída do gerador de números aleatórios até a entrada A1 do bloco de identificação. Dessa maneira os pacientes que entrarem no modelo serão classificados aleatoriamente em um dos quatro subgrupos. O resultado depois desses passos pode ser visto na Figura 18.

Se for necessário, os blocos podem trocar de lugar arrastando-os de acordo. As linhas que representam as ligações entre os blocos também podem ser movidas da mesma forma. Ainda, os blocos e as linhas podem ser removidos do modelo selecionando-os e pressionando a tecla *Delete*.

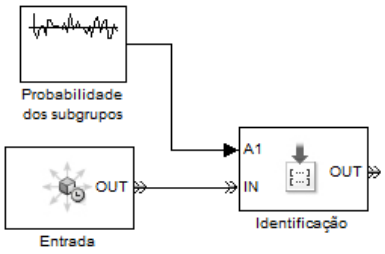

Figura 18 - Representação do modelo após as ligações entre os blocos.

4.3.5. Filas de espera

As filas de espera são uma maneira de controlar o fluxo de pacientes pelo modelo quando estamos representando a utilização de algum recurso escasso. Os pacientes que entram na fila precisam aguardar até que alguma condição seja satisfeita para que possam avançar.

A ordem de saída dos pacientes pode depender da ordem de chegada ou de alguma característica como, por exemplo, a gravidade do estado de saúde na qual ele se encontra. Vamos adicionar logo após o bloco de identificação uma fila de espera do tipo *first-in-first-out*, ou "primeiro a entrar, primeiro a sair", que é representada pelo bloco *FIFO Queue* da biblioteca S*imEvents Queues*. Nas configurações do bloco podemos escolher o número máximo de pacientes que podem permanecer na fila. Neste caso vamos remover este limite, alterando a capacidade para "Inf". Após a criação do bloco "Fila de espera" é preciso ligá-lo ao bloco "Identificação". O resultado depois desses passos pode ser visto na Figura 19.

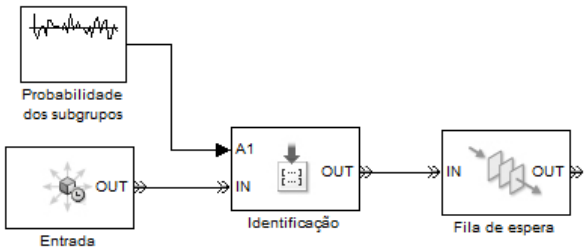

Figura 19 - Representação do modelo após a inclusão da fila de espera.

A fila de espera por si só serve apenas como um lugar para armazenar as entidades. Para limitar a passagem dos pacientes precisamos incluir uma barreira, representada por um bloco *Gate*, disponível na biblioteca *SimEvents Gates*. Existem dois tipos de barreiras, a *Enabled gate* e a *Release gate*. A primeira pode ser aberta ou fechada de acordo com algum evento, e permanece assim até que outro evento altere seu estado. A outra permanece fechada até que um evento faça com que ela seja aberta, e então apenas um paciente é liberado e a barreira se fecha novamente.

Neste exemplo iremos utilizar a *Enabled gate*. Essa barreira é aberta quando um sinal contendo um número positivo chega até a porta "en" e é fechada quando o número for negativo ou igual a zero. Após a criação do bloco *Enabled gate*, chamado de "Limite de atendimentos" na Figura 20, liga-se a sua entrada *IN* à saída *OUT* do bloco Fila de espera.

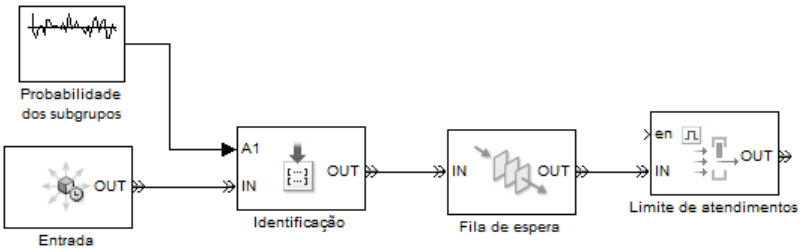

Figura 20 - Representação do modelo com a fila de espera e a barreira.

Para controlar o número de atendimentos será utilizado um bloco *Entity Departure Counter*, que se encontra na biblioteca *SimEvents Probes*. Esse bloco, que será chamado "Pacientes no dia", faz a contagem do número de pacientes que passam por ele. Após a criação desse bloco liga-se a sua entrada *IN* à saída do bloco "Limite de atendimentos".

A configuração padrão desse bloco cria um atributo com o valor da contagem. Neste modelo é necessário fazer com que o número seja enviado através de uma nova porta chamada "#d". Isto é feito na janela de configurações do bloco alterando a opção *Write count to signal port #d* para *On*, e alterando a opção *Write count to attribute* para *Off*. Também é preciso fazer com que a contagem seja reiniciada a cada dia, escolhendo a opção *Reset counter upon*: *Sample time hit from port ts*. A janela de configurações com as opções utilizadas neste exemplo é mostrada na Figura 21. O resultado depois desses passos pode ser visto na Figura 22.

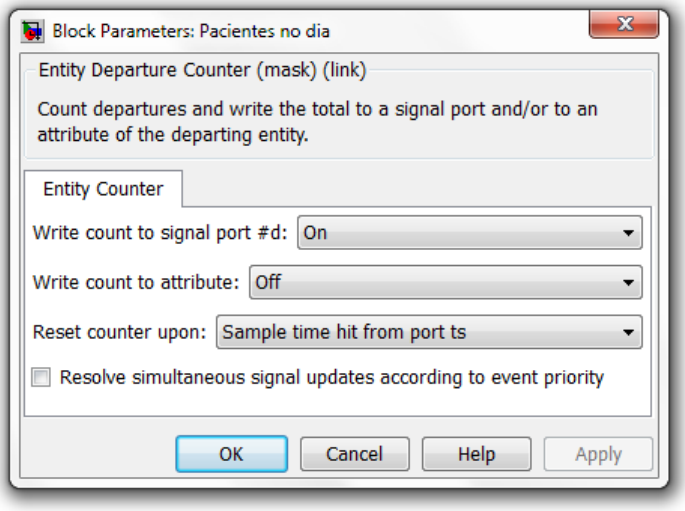

Figura 21 - Contador para o número de pacientes atendidos por dia.

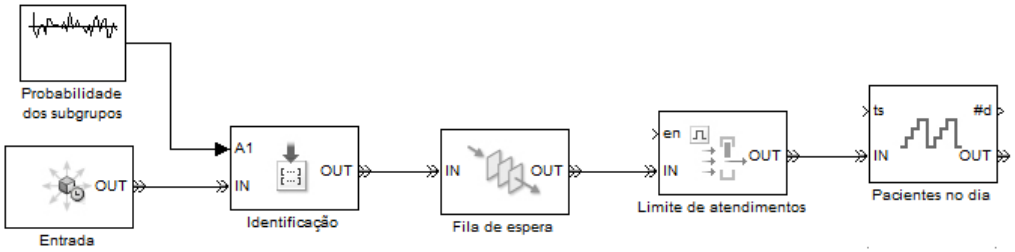

Figura 22 - Representação do modelo após a inclusão e configuração do bloco Pacientes no dia.

A opção *Sample time hit* faz com que a contagem seja reiniciada toda vez que um novo sinal chegar até a porta "ts". Para enviar esse sinal no início de cada dia vamos incluir um bloco *Constant* da biblioteca *Simulink Sources*, que será chamado "Novo dia". Esse bloco possui em suas configurações uma opção chamada *Sample time* que controla de quanto em quanto tempo o bloco deve enviar um novo sinal. Ligando a saída deste bloco à porta ts do bloco "Pacientes no dia" e alterando a opção *Sample Time* para 1, a contagem de pacientes será reiniciada todos os dias. A Figura 23 mostra o modelo após esse passo.

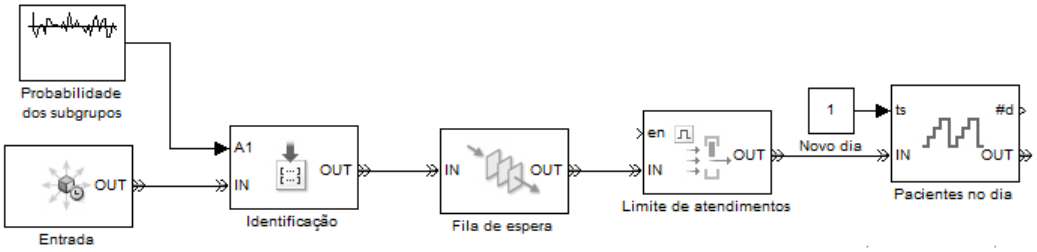

Figura 23 - Representação do modelo após a inclusão do bloco Novo dia.

4.3.6. Subsistemas

O próximo passo é checar se o número máximo de pacientes atendidos durante um dia foi atingido e, se for o caso, fazer com que os próximos pacientes tenham que aguardar na fila até o dia seguinte. Ou seja, contar o número de pacientes já atendidos no dia e comparar esse número com o número máximo de atendimentos por dia.

Para isso, é preciso trabalhar com o sinal que contém a contagem dos pacientes utilizando blocos da biblioteca *Simulink*. Porém, os blocos dessa biblioteca não foram feitos para a simulação de eventos discretos e podem não funcionar conforme o esperado, uma vez que eles não dependem da ocorrência de um evento para serem executados. Para corrigir isso se utiliza um subsistema de eventos discretos, que faz com que os blocos que ele contém sejam executados somente quando um evento provocar a sua execução. Esse bloco, que será chamado de "Verifica o limite", está na biblioteca *SimEvents Ports and Subsystems* com o nome de *Discrete Event Subsystem*, e representa um subsistema. Com um duplo clique no subsistema é aberta uma nova janela onde se pode ver e alterar o seu conteúdo.

A Figura 24 mostra o bloco do subsistema com as portas de entrada e saída trocadas. Essa troca pode ser feita clicando uma vez sobre o bloco para selecioná-lo, e então escolhendo a opção *Format → Flip Block*.

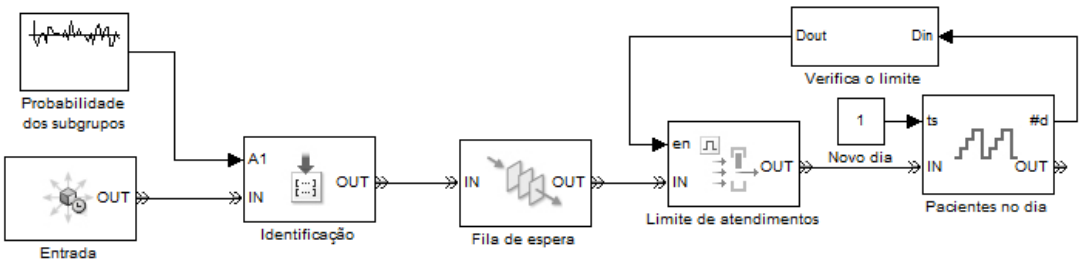

Figura 24 - Representação do modelo com a inclusão do subsistema que controla a barreira.

Para fazer a comparação utiliza-se o bloco *Compare to Constant* da biblioteca S*imulink Logic and Bit Operations*. Esse bloco compara o valor de um sinal com uma constante predeterminada e retorna o valor 1 se a comparação for verdadeira e 0 se for falsa. Por padrão, o bloco executa a comparação menor ou igual ("<="), ou seja, compara se o sinal é menor ou igual ao valor da constante, que é a comparação que precisa ser feita neste exemplo, pois queremos que o valor seja 1 sempre que o limite não tiver sido atingido e 0 quando o valor passar do limite. Se for necessário, essa comparação pode ser trocada através da opção *Operator* nas configurações do bloco. Esse bloco foi incluído dentro do subsistema "Verifica o limite" com o nome "Compara com limite", como pode ser visto na Figura 25.

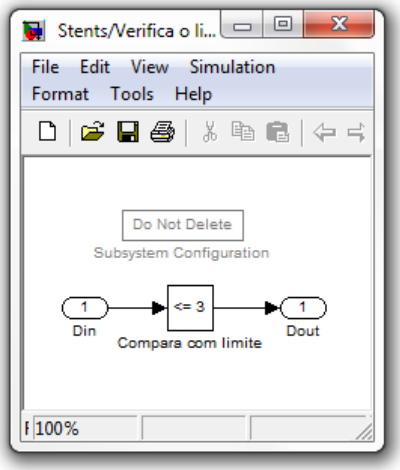

Figura 25 - Janela do subsistema Verifica o limite após a inclusão do bloco Compara o limite.

Neste ponto é necessário definir o valor da constante, que deve ser igual ao limite que estamos querendo comparar. Neste exemplo, esse limite varia conforme a capacidade que está sendo testada. Por esse motivo é indicado utilizar uma variável para definir esse limite. Nesse

caso o valor a ser comparado é definido como sendo o nome de uma variável, por exemplo, limite. Isso é feito alterando a opção *Constant Value* nas configurações do bloco "Compara com limite" para "limite", sem aspas. O valor dessa variável é controlado pela janela de comandos do MATLAB, e precisará ser definido antes de o modelo ser executado.

Também é preciso adicionar um bloco chamado *Data Type Conversion*, da biblioteca *Simulink Signal Attributes*, pois o sinal criado pelo bloco que executa a comparação é incompatível com o tipo de sinal aceito pela barreira. Este bloco executa esta conversão automaticamente e, portanto, não é necessário alterar nenhuma de suas configurações.

O subsistema "Verifica o limite" completo pode ser visualizado na Figura 26.

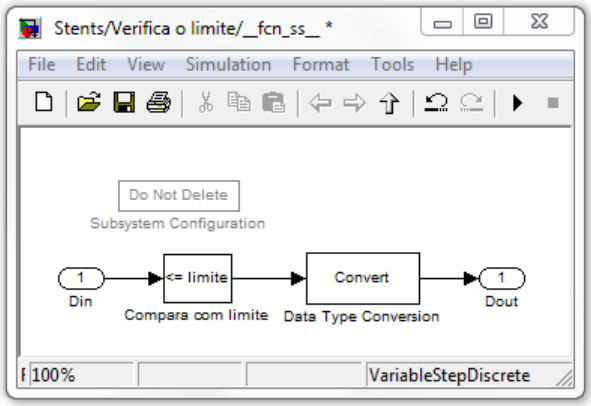

Figura 26 - Subsistema discreto que controla o número de atendimentos por dia.

Conforme mais blocos são incluídos ao modelo, mais difícil se torna organizá-los. Para facilitar essa organização existe outro tipo de subsistema que pode ser utilizado. Este subsistema está na biblioteca *Simulink Ports and Subsystems* e é utilizado em conjunto com o bloco *Conn* da biblioteca *Simevents* para fornecer uma entrada e saída para as entidades.

Uma maneira mais fácil de criar um subsistema deste tipo é simplesmente selecionar todos os blocos que farão parte do subsistema e escolher a opção *Edit* → *Create Subsystem*, pois assim as entradas e saídas são configuradas automaticamente. Para selecionar mais de um bloco pode-se utilizar a tecla *shift* do teclado e clicar em cada um dos blocos, ou então clicar e arrastar o mouse em um ponto vazio da janela do modelo, criando um retângulo que selecionará todos os blocos contidos em seu interior.

O número das portas de entrada e saída do subsistema pode mudar a cada vez que esse é criado, mas isso não influencia em nada no funcionamento do modelo. O tamanho do bloco do subsistema pode ser alterado selecionando o bloco e então clicando e arrastando em um

dos quatro quadrados que aparecem em seus cantos. O mesmo pode ser feito para os outros blocos do modelo.

A Figura 27 mostra o modelo depois que foi criado o subsistema “Fila de espera” com os blocos que controlam a fila de espera, e a Figura 28 mostra o conteúdo desse subsistema.

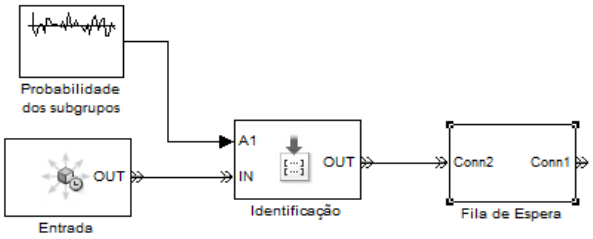

Figura 27 - Representação do modelo após a criação do subsistema para os blocos da fila e espera.

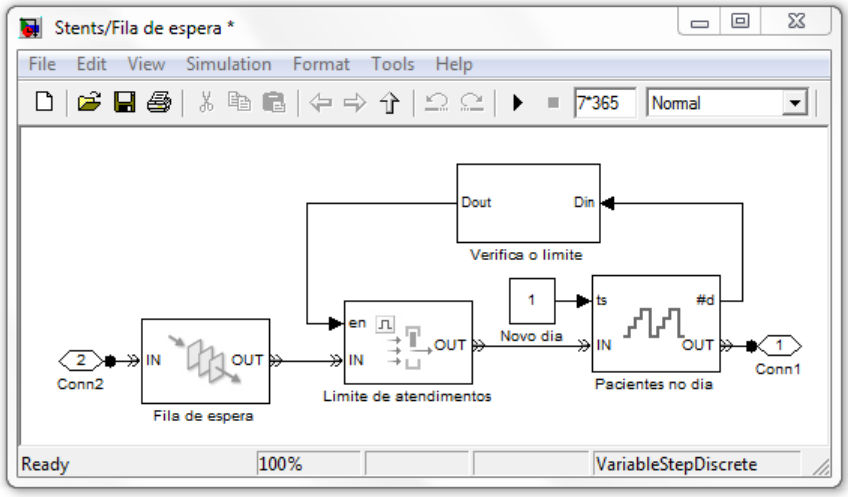

Figura 28 - Janela do subsistema "Fila de espera".

4.3.7. Prioridades

Como a simulação de eventos discretos trabalha com eventos que ocorrem ao longo do tempo, pode acontecer de dois ou mais deles ocorrerem simultaneamente. A ordem na qual eles serão computados pode fazer diferença no resultado final do modelo. Essa ordem é controlada pela prioridade atribuída a cada um dos eventos e pode ser alterada conforme for necessário, com exceção de alguns eventos que são controlados pelo sistema. Mesmo assim, algumas vezes é preciso incluir no modelo um bloco que seja capaz de armazenar uma entidade durante certo tempo, até que outros eventos e cálculos sejam executados.

Por padrão a simulação avalia primeiramente os eventos que possuem o menor valor de prioridade, mas essa opção pode ser alterada na janela de configurações da simulação para que sejam avaliados primeiramente os eventos com maior valor, se for conveniente.

Neste exemplo, os pacientes que estiverem aguardando na fila serão liberados de uma só vez quando um novo dia iniciar, podendo ocasionar problemas no modelo. Para prevenir estes problemas, adiciona-se um bloco do tipo *Server*, disponível na biblioteca *SimEvents Servers*, que armazena as entidades por um tempo determinado. Existem 3 variações desse bloco, cada uma podendo armazenar simultaneamente um diferente número de entidades. Aqui será utilizado o *Infinite Server*, que não possui esse limite e será incluído dentro do subsistema "Fila de espera". Este novo bloco é chamado de "Espera Fila" e pode ser visualizado na Figura 29.

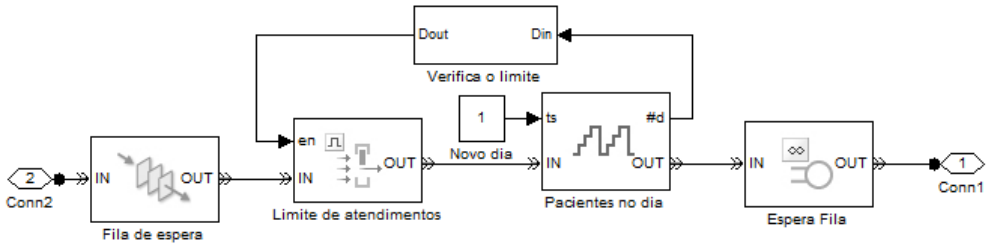

Figura 29 - Representação da fila de espera após a inclusão do bloco *Server*.

Nas configurações desse bloco é preciso alterar o tempo de serviço para 0, pois é preciso que o bloco armazene a entidade somente enquanto os eventos simultâneos ocorrem.

Outra das opções desse bloco é a *Service completion event priority*, que controla a sua prioridade. Todos os blocos *Server* possuem por padrão valor de prioridade igual a 500. Neste exemplo os pacientes que saem da fila de espera são encaminhados para receber o tratamento, que também irá envolver um bloco *Server*, e é preciso que cada paciente passe completamente por esse *Server* antes de o próximo paciente ser encaminhado até ele, para que não ocorram problemas. Para garantir que isso aconteça altera-se a prioridade do bloco "Espera Fila" para qualquer valor maior que 500, como 600, por exemplo.

Assim, os pacientes serão liberados da fila de espera somente depois de serem executados todos os eventos que possuírem prioridade menor que 600, permitindo ao modelo calcular corretamente os próximos eventos para cada paciente.

4.3.8. Modificando atributos

Agora que a fila de espera está construída, é preciso calcular o tempo que cada paciente permanece aguardando na fila e o custo associado a essa espera. A biblioteca *SimEvents Timing* possui dois blocos chamados *Start Timer* e *Read Timer* que servem para essa finalidade. O primeiro define o momento a partir do qual se quer calcular o tempo, enquanto o segundo calcula o tempo decorrido daquele momento até o momento presente.

Cada um desses blocos é associado a uma variável chamada *Timer Tag* que define o momento inicial. Um bloco *Start Timer* deve ser incluído antes de o paciente entrar na fila de espera e um bloco *Read Timer* logo após a sua saída, como mostra a Figura 30.

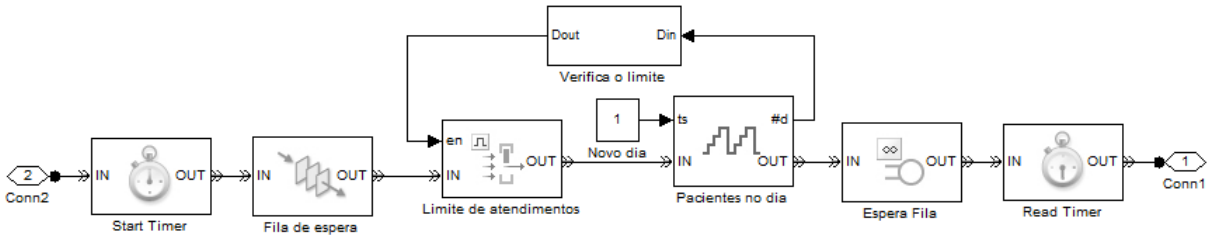

Figura 30 - Representação do subsistema "Fila de espera" após a inclusão dos *Timers*.

Por padrão os dois blocos recém criados são associados a uma *Timer Tag* chamada "*T1"*, mas nas configurações desses blocos é possível alterar o nome dessa *Timer Tag*, desde que ele seja igual para os dois blocos.

Configura-se também o bloco *Start Timer* para reiniciar a contagem se a *Timer Tag* já tiver sido definida, alterando em suas configurações a opção *Warn and Continue* para *Restart*, e habilita-se a porta *Elapsed time* na aba *Statistics* das configurações do bloco *Read Timer*, que calcula o tempo decorrido, alterando a opção *Elapsed time, et* para *On*.

O próximo passo é criar um atributo que contenha o custo associado a cada paciente. No bloco de identificação na qual definimos o subgrupo anteriormente adicionamos um novo atributo "Custo" com valor inicial igual a 0. Isto é feito abrindo a janela de configurações do bloco "Identificação" e clicando no primeiro dos três botões à esquerda da janela. Será criado um novo atributo chamado "*Attribute2*", que deve ser alterado para "Custo", e o seu valor deve ser definido como 0.

Para alterar o valor do custo conforme o tempo na fila precisamos primeiramente ler o valor atual através de um bloco chamado *Get Attribute*, mudar esse valor e salvá-lo novamente com um bloco *Set Attribute*, ambos disponíveis na biblioteca *SimEvents Attributes*. Primeiramente, adiciona-se no subsistema "Fila de espera" um bloco *Get Attribute*, que será chamado "Custo atual", configurando-o para ler o atributo "Custo", que é feito trocando o nome "*Attribute1*" para "Custo" em suas configurações. Feito isso, adiciona-se também o bloco *Set Attribute*, que será chamado "Novo custo", e em suas configurações troca-se o nome "*Attribute1*" para "Custo" e altera-se a opção "*Value from dialog*" para "*Value from signal port*".

Novamente é preciso utilizar um subsistema de eventos discretos, *Discrete Event Subsystem* da biblioteca *SimEvents Ports and Subsystems*, que será chamado "Calcula o custo", para incluir os blocos que irão alterar o custo e também de um *Server* para armazenar a entidade enquanto os cálculos são efetuados. Dessa vez será utilizado um *Single Server*, da biblioteca S*imEvents Servers*, para garantir que apenas uma entidade seja armazenada enquanto o custo de espera é calculado. O tempo de serviço desse bloco precisa ser alterado em suas configurações para ser igual a 0, pois enquanto o computador está efetuando os cálculos o tempo fica parado, ou seja, o bloco precisa segurar o paciente somente enquanto os cálculos estão sendo feitos e que o libere sem deixar que o tempo passe. A Figura 31 mostra o subsistema "Fila de espera" após a execução desses passos.

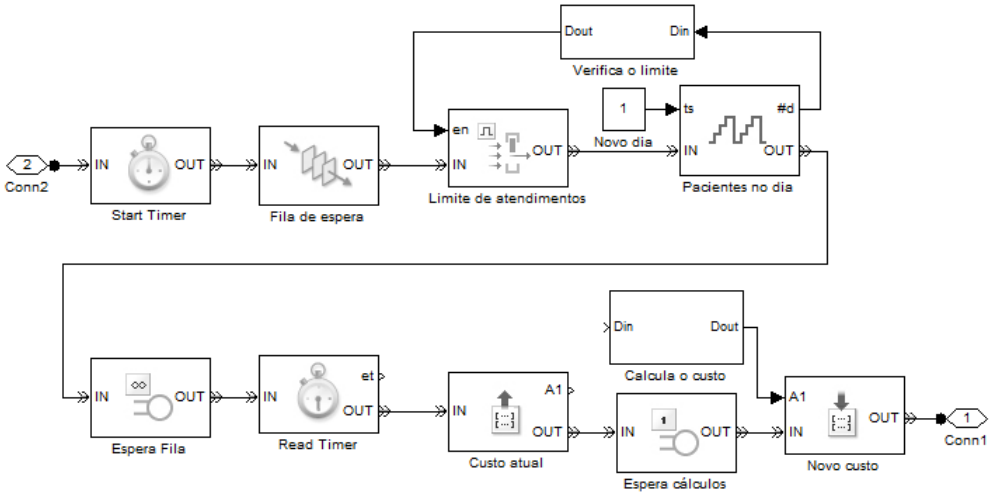

Figura 31 - Representação da fila de espera com os blocos para calcular o custo.

Para poder calcular o custo, é preciso adicionar uma nova porta de entrada *Din* ao subsistema "Calcula o custo", que também pode ser encontrada na biblioteca *SimEvents Ports and Subsystems*.

Depois de ser adicionada essa nova porta, no subsistema "Fila de espera" conecta-se uma das entradas do subsistema "Calcula o custo" à saída "*et*" do bloco "Read timer" e a outra entrada à saída "A1" do bloco "Custo atual", conforme é mostrado pela Figura 32.

Para multiplicar o tempo pelo custo de espera utiliza-se o bloco *Gain*, que multiplica um valor por uma constante fixa, e o bloco *Sum* para adicionar esse valor ao custo, ambos disponíveis na biblioteca *Simulink Math Operations*. Os dois blocos devem ser adicionados ao subsistema "Calcula o custo", e é preciso alterar o valor do bloco *Gain*, que é chamado "Custo espera", para 15, que é o custo de um dia de espera, e no bloco *Sum* alterar a lista de sinais para "++" fazendo com que os dois valores sejam somados.

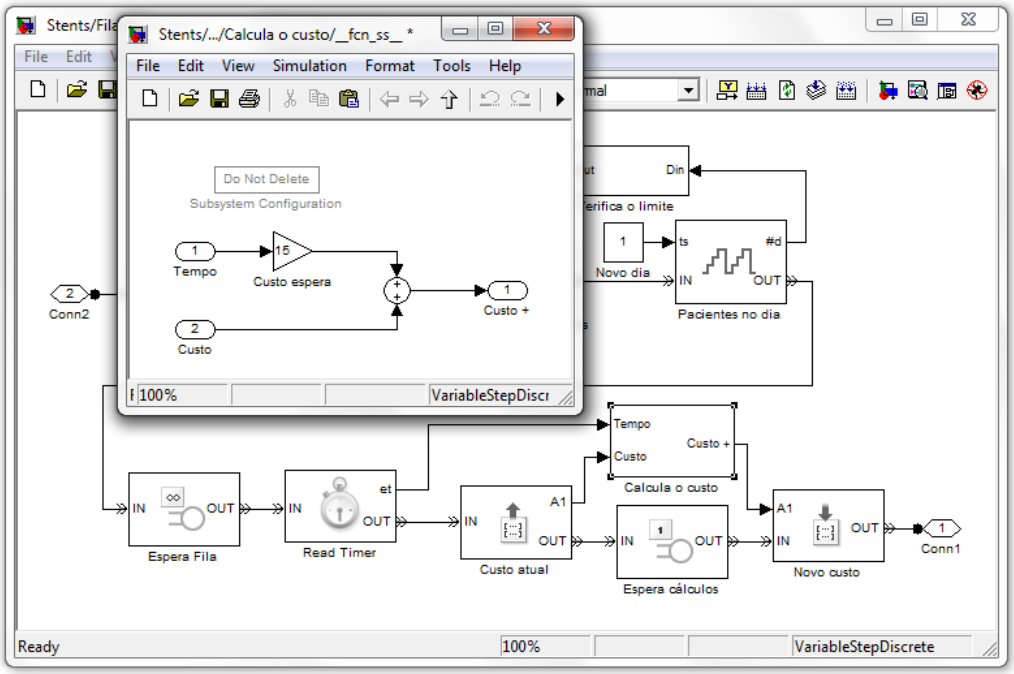

Figura 32 - Subsistema discreto para calcular o custo de espera.

Agora é preciso incluir o número de intervenções realizadas por cada paciente e o custo de cada uma delas da mesma forma que foi feito para o custo da fila de espera. Para armazenar a quantidade de intervenções a que o paciente foi submetido cria-se um novo atributo “Interv” no bloco “Identificação” com valor igual a 0.

Na janela do modelo, adiciona-se um bloco *Get Attribute* para ler os atributos, que será chamado “Lê Atributos”, e um bloco *Set Attribute* para salvar os novos valores desses atributos, que será chamado “Salva Atributos”. Esses dois blocos estão disponíveis na biblioteca *SimEvents Attributes*.

Também precisa ser adicionado um *Single Server* e um subsistema de eventos discretos, do mesmo modo que foi feito para a fila de espera. Precisam ser lidos pelo bloco “Lê Atributos” os atributos “Subgrupo”, “Custo” e “Interv”, e o bloco “Salva Atributos” deve salvar o novo custo e o número de intervenções novamente nos atributos “Custo” e “Interv”. Para isso o subsistema precisa ter 3 portas de entrada e 2 portas de saída, e essas portas precisam ser ligadas ás respectivas portas dos blocos “Lê Atributos” e “Salva Atributos”.

A Figura 33 mostra a representação do modelo com esses quatro blocos e as ligações que devem ser feitas entre eles.

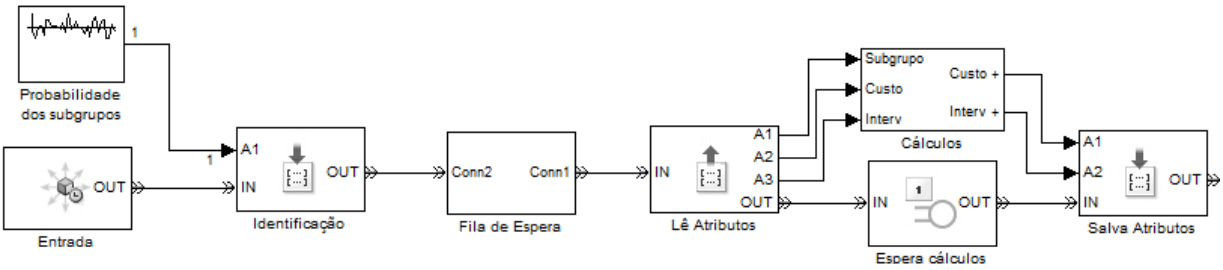

Figura 33 - Representação do modelo após a inclusão dos blocos que representam a intervenção.

O subsistema "Cálculos" será utilizado para alterar o custo e o número de intervenções de cada paciente. Para adicionar uma unidade ao número de intervenções pode-se utilizar o bloco *Bias*, da biblioteca *Simulink Math Operations*. Esse bloco adiciona uma constante ao valor atual de um sinal. Neste exemplo essa constante terá valor 1.

Para calcular o custo da intervenção não é possível utilizar esse mesmo bloco, pois o custo depende do subgrupo em que o paciente é classificado. Para separar estes valores é preciso do bloco *Multiport Switch* da biblioteca *Simulink Signal Routing*. Este bloco recebe um número referente a uma de suas portas de entrada e repassa o valor vindo desta porta ao próximo bloco.

Os custos da intervenção com cada um dos tipos de stents são atribuídos através de um bloco *Constant* com o valor correspondente e *sample time* igual a "inf". Esses blocos podem ser ligados manualmente a cada porta do bloco M*ultiport Switch*, ou pode-se utilizar uma variável para definir as ligações, o que facilita muito quando forem testadas as diferentes estratégias. Para fazer isso, é preciso incluir três blocos da biblioteca *Simulink Signal Routing* ao modelo.

Primeiro, utiliza-se um bloco *Mux* para agrupar os dois sinais contendo os custos, criando assim um novo sinal que contém um vetor com os dois valores para serem utilizados pelo bloco *Selector*. Esse bloco recebe um vetor de dados, reordena os valores deste conforme alguma regra, e retorna esse novo vetor.

Nas opções do bloco *Selector*, define-se o *Input port size* como 2, que é o número de valores no vetor original. A regra definida por padrão é a *Index vector (dialog)*, que reordena os valores conforme um novo vetor que é especificado. Por exemplo, se for utilizado o vetor [2 1 2 1] para definir a regra, será criado um novo vetor com o segundo valor do vetor original nas posições 1 e 3, e com o primeiro valor nas posições 2 e 4.

Assim, é possível definir uma variável para especificar qual custo que será associado a cada subgrupo, alterando a opção *Index* de [1 3] para o nome de uma variável, como CustoSub, por exemplo.

Depois de definir corretamente como será criado o vetor correto, utiliza-se um bloco *Demux* para separar os valores em quatro sinais, que serão ligados às portas correspondentes no bloco *Multiport Switch*, como mostra a Figura 34. Por fim, os blocos referentes à intervenção também são separados em um subsistema para facilitar a organização, que pode ser visto na Figura 35.

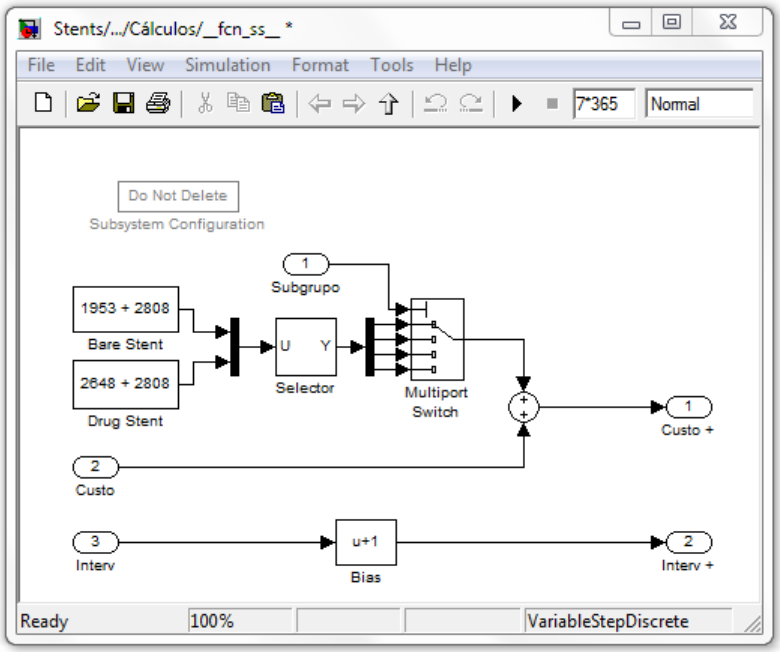

Figura 34 - Subsistema discreto "Cálculos" que executa o cálculo do custo da intervenção.

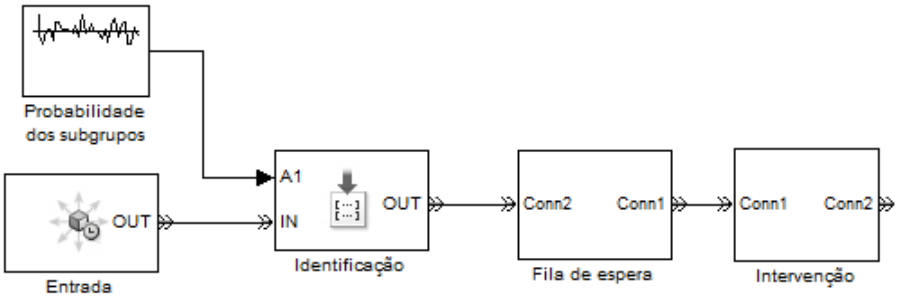

Figura 35 - Representação do modelo após a criação do subsistema "Intervenção".

4.3.9. Controlando caminhos

Após os pacientes serem tratados, existe a possibilidade de a doença se manifestar novamente, fazendo com que precisem de uma nova intervenção. Neste exemplo considera-se que se isso não ocorrer no período de um ano após o tratamento, o paciente estará curado e sairá do modelo.

O tempo até a recorrência é definido por uma distribuição exponencial com taxas diferentes para cada um dos subgrupos. Para incluir isso ao modelo, adiciona-se após o subsistema que representa a intervenção um bloco *Output Switch*, da biblioteca *SimEvents Routing*. Este bloco separa o caminho dos pacientes de acordo com algum critério, que neste caso será o atributo "Subgrupo". Para isso, altera-se o número de portas para 4, escolhe-se a opção *Switching criterion: From attribute* e define-se "Subgrupo" como o nome do atributo nas configurações desse bloco.

Para separar os valores de cada subgrupo não é possível utilizar novamente o bloco *Multiport Switch*, pois ele não é compatível com o bloco gerador de números aleatórios. Neste caso, é preciso utilizar um gerador de números aleatórios para atribuir o tempo de recorrência a um atributo "TempoReoc" em cada um dos quatro caminhos, com cada um desses geradores seguindo uma distribuição exponencial com média definida de acordo com a estratégia sendo testada. Aqui também é possível utilizar uma variável para definir a média da exponencial em cada um dos geradores. A Figura 36 mostra um novo subsistema "Recorrência", que foi criado para conter os blocos descritos até agora.

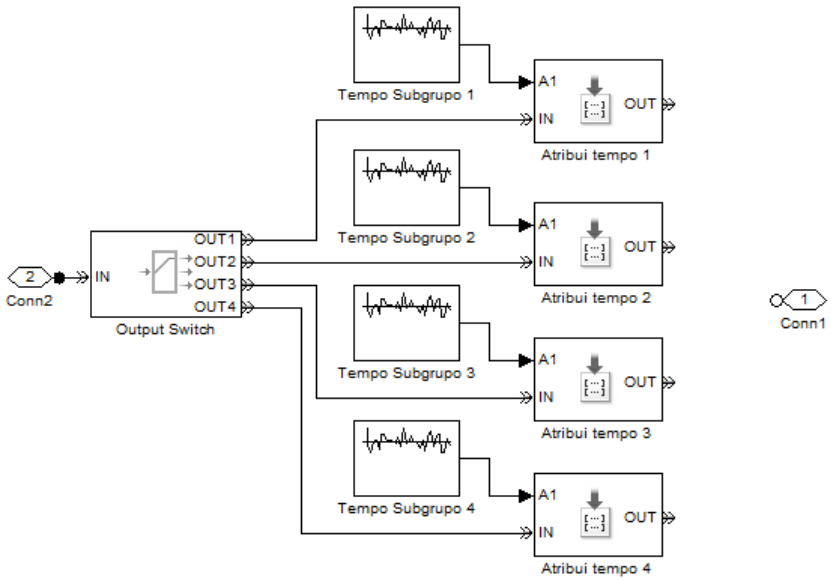

Figura 36 - Subsistema que controla a atribuição do tempo de recorrência.

Após o valor ter sido atribuído é preciso juntar novamente todos os pacientes em um mesmo caminho através de um bloco *Path Combiner*, disponível na biblioteca *SimEvents Routing*. Para que os pacientes que não sofrerem novamente com a doença sejam removidos do modelo após um ano, é preciso utilizar o bloco *Schedule Timeout* da biblioteca *SimEvents Timing*. Este bloco permite que seja definido um tempo máximo que uma entidade pode permanecer em um bloco. Após esse tempo a entidade é removida do bloco por uma porta diferente das outras entidades.

No subsistema "Recorrência", adiciona-se o bloco *Schedule Timeout*, e em suas configurações altera-se o *Timeout Interval* para 365 dias. Depois, adiciona-se um *Infinite Server* para armazenar as entidades até acontecer a recorrência ou o *Timeout*. Para isso, escolhe-se a opção *Service time from: Attribute*, com o nome do atributo "TempoReoc", e habilita-se na aba *Timeout* a porta TO. Os pacientes que precisarem realizar uma nova intervenção irão sair pela porta OUT, que deve estar ligada a um bloco *Cancel Timeout* para que seja cancelado o *timeout* definido anteriormente. A Figura 37 mostra o subsistema "Recorrência" após a inclusão desses blocos.

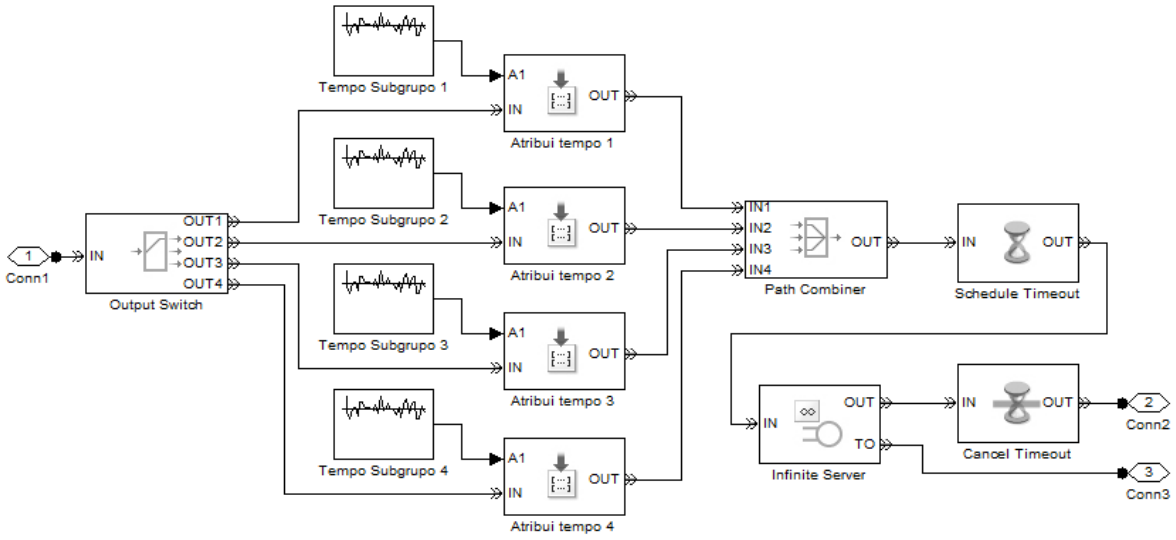

Figura 37 - Subsistema para o tempo de recorrência com a separação do caminho dos pacientes.

Agora é preciso verificar quantas intervenções cada paciente já realizou para que ele receba a intervenção correta. Para isso, utiliza-se mais um bloco *Outport Switch* para separar os pacientes em dois caminhos de acordo com o atributo "Interv". Os pacientes que fizeram somente uma intervenção precisam ser colocados de novo na fila de espera juntamente com os novos pacientes. Para fazer isso é utilizado um bloco *Path Combiner*, disponível na biblioteca *SimEvents Routing*. A Figura 38 mostra o modelo após esses passos.

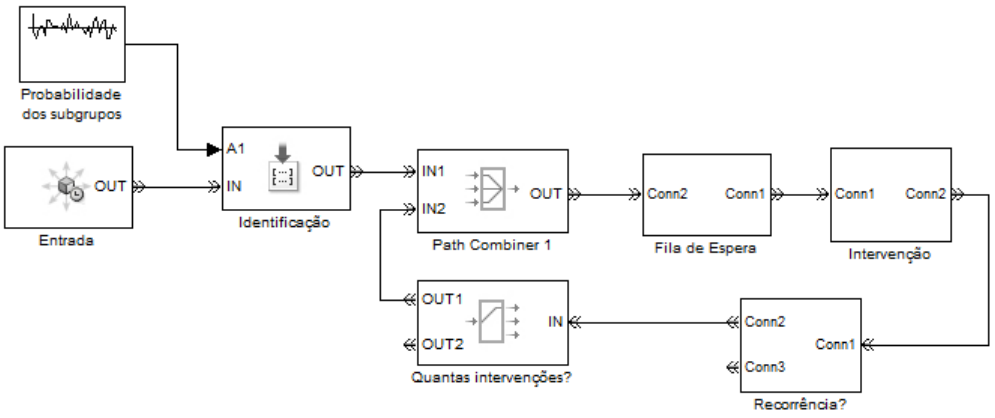

Figura 38 - Modelo após a inclusão da recorrência e do número de intervenções.

Os pacientes que já tiverem realizado duas intervenções são encaminhados para receber uma cirurgia de revascularização miocárdica. A representação da cirurgia é bastante parecida com a intervenção, assim, é possível copiar o subsistema que contém a intervenção e alterar o que for preciso. Isso pode ser feito selecionando o bloco do subsistema "Intervenção", e escolhendo as opções *Edit* → *Copy* e *Edit* → *Paste*. Como todos os pacientes recebem a mesma cirurgia, não é mais necessário saber o subgrupo, e podemos utilizar um bloco *Bias* para alterar o custo. A representação completa desse subsistema, que é chamado "Cirurgia", pode ser visualizada na Figura 39.

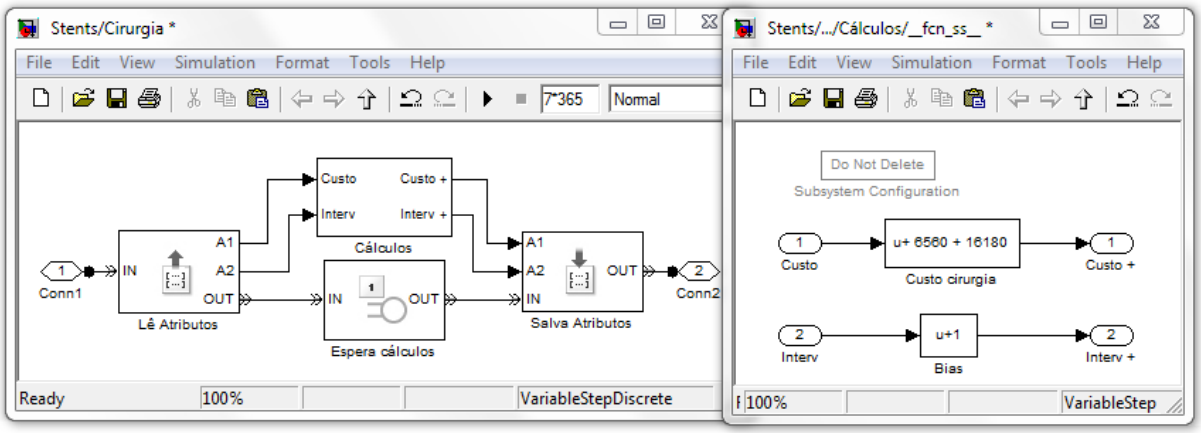

Figura 39 - Subsistema que calcula o custo da cirurgia.

4.3.10. Qualidade de vida

O cálculo que será utilizado para calcular a qualidade de vida média já foi discutido anteriormente. Para realizar esse cálculo é preciso contar todo o tempo que o paciente permanece no modelo, e associar corretamente a qualidade de vida a cada período de tempo, para depois poder calcular a qualidade média durante o tratamento.

No modelo foi criado, o tempo passa para os pacientes somente quando estes estão na fila de espera ou quando ainda não aconteceu uma recorrência. Considera-se que todos os pacientes que sofrem com a doença possuem igual qualidade de vida até serem tratados. Para incluir isso ao modelo é preciso simplesmente multiplicar o tempo que eles aguardam na fila pela qualidade de vida associada ao *baseline*.

Primeiramente define-se um novo atributo chamado “Qualidade” com valor igual a 0 no bloco “Identificação”. Feito isso, modifica-se o subsistema que contém a fila de espera para incluir a qualidade durante o tempo de espera.

Feito isso, é preciso incluir o atributo “Qualidade” nos blocos *Get* e *Set Attribute*, criar uma nova porta de entrada e outra de saída no subsistema discreto, e somar o tempo de espera multiplicado pela qualidade no *baseline* ao valor atual da qualidade. Isso é feito facilmente utilizando os blocos *Gain* e *Sum* da mesma forma que foi feito para o custo de espera, como é mostrado pela Figura 40.

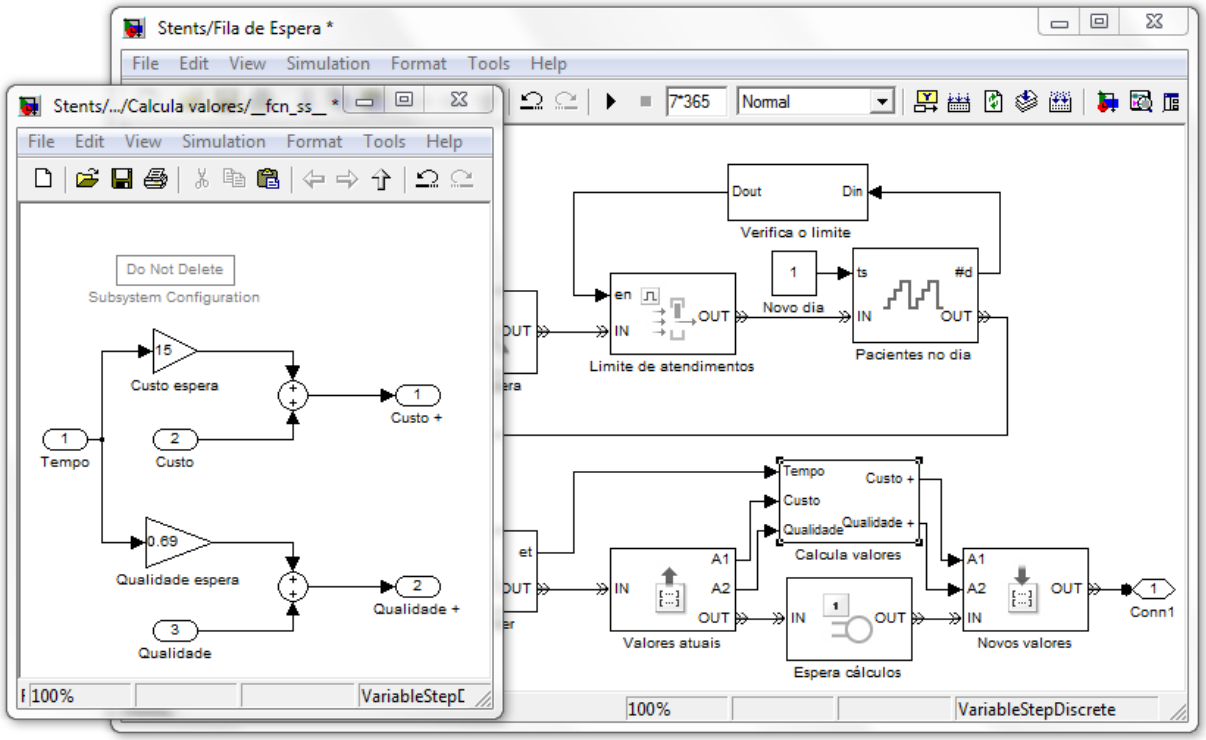

Figura 40 - Inclusão do cálculo da qualidade de vida durante a espera.

Agora, a qualidade de vida dos pacientes que receberam a intervenção aumenta no período de um ano, a não ser que ele sofra novamente com a doença. A forma com que esta conta é feita já foi descrita anteriormente como sendo a área sobre a curva formada pela qualidade ao longo do tempo.

A maneira mais fácil de incluir isso no modelo é utilizando o bloco *Attribute Function*, que permite que seja criada uma função que utilize o valor de um ou mais atributos e modifique-os através de funções disponíveis no MATLAB. O bloco se encontra na biblioteca *SimEvents User-Defined Functions* e precisa que o MATLAB possua um compilador C configurado para poder funcionar. Este bloco deve ser incluído dentro do subsistema que controla a recorrência, no local indicado pela Figura 41.

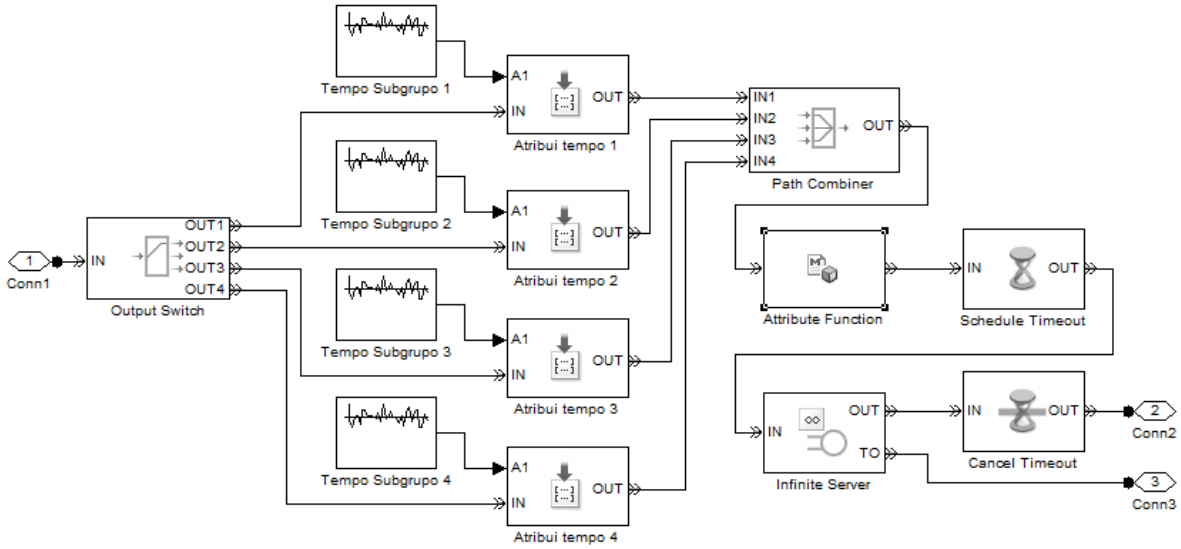

Figura 41 - Inclusão do cálculo da qualidade de vida após a intervenção.

Neste exemplo, precisa ser utilizada uma função que encontre o valor de um ponto intermediário em uma linha definida por dois pontos, e outra função para calcular a área sob a curva formada por um conjunto de pontos. Essas funções são chamadas *interp1* e *trapz*, respectivamente.

Com um duplo clique no bloco *Attribute Function* uma nova janela chamada *MATLAB Function Block Editor* é aberta. É nesta janela que será criado o código que fará o cálculo da qualidade de vida. Uma maneira de escrever este código é demonstrada no Apêndice A.

Depois de incluir as medidas de qualidade, é preciso contar o tempo total que o paciente permanece no modelo. Isso é feito facilmente adicionando um bloco *Start Timer* logo após a identificação do paciente.

Para calcular a qualidade média durante o tempo de permanência no modelo primeiramente deve-se juntar os caminhos de todos os pacientes que estiverem saindo do modelo, ou seja, aqueles que ficaram bem por um ano e aqueles que receberam a cirurgia, utilizando o bloco *Path Combiner*.

Logo após este bloco cria-se um novo subsistema contendo os blocos *Read Timer*, para saber o tempo total, e os blocos *Get* e *Set Attribute*, bem como um *Single Server* e um subsistema discreto para realizar o cálculo. Dentro do subsistema discreto é preciso utilizar somente um bloco *Divide* para realizar a divisão da qualidade total pelo tempo de permanência no modelo.

Por padrão, o subsistema discreto é executado toda vez que um sinal chega a qualquer uma de suas portas, ou seja, um subsistema com duas portas como este será executado duas vezes, e o resultado final será o obtido após a segunda execução.

Dependendo da ordem com que o programa executar o subsistema, pode ser que o valor do tempo ainda não tenha sido atualizado quando a primeira conta for feita, gerando uma mensagem de erro, pois será efetuada uma divisão por zero.

Para corrigir isso é preciso alterar as prioridades das portas de entrada do subsistema discreto, chamadas de "Tempo" e "Qualidade" na Figura 42. Isso é feito marcando a opção *Resolve simultaneous signal updates according to event priority* e diminuindo o valor de prioridade nas configurações da porta que recebe o tempo, fazendo com que ela seja atualizada antes que a porta que recebe a qualidade.

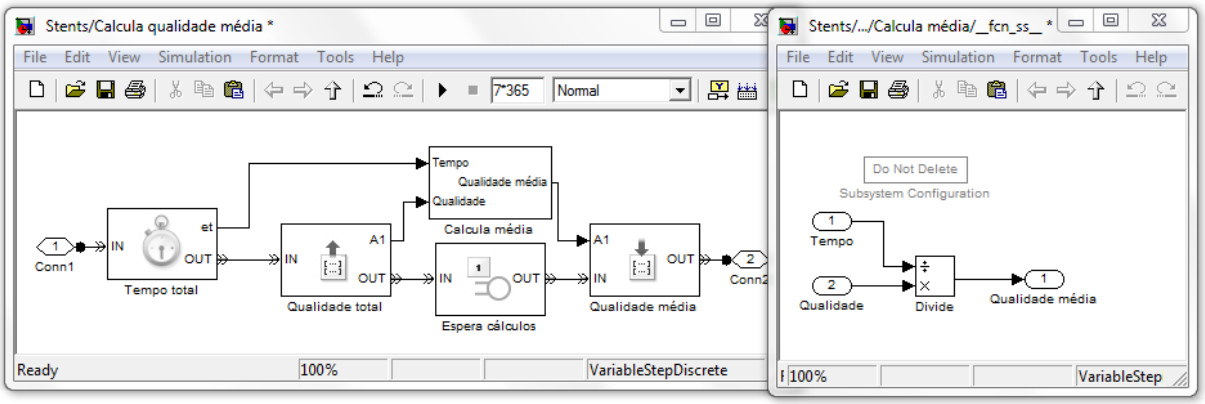

Figura 42 – Subsistema utilizado para calcular a qualidade de vida média.

4.3.11. *Burn-in*

Para executar a simulação corretamente, deve-se pensar no impacto da configuração inicial sobre os resultados finais. Neste exemplo, no início da simulação não existe nenhum paciente no modelo. Isso pode alterar o resultado obtido no decorrer da simulação, uma vez que o número de pacientes na fila influencia o resto dos pacientes, bem como o número de pacientes que já receberam uma intervenção e que podem precisar entrar novamente na fila.

Uma maneira de diminuir os efeitos dessa configuração inicial é utilizar um período no início da simulação em que não será observado nenhum resultado, chamado de *Burn-in*. Assim, os primeiros pacientes servirão somente para popular o modelo, e não influenciarão os resultados finais. Os pacientes que entrarem após esse período serão tratados normalmente, e ao final servirão para obter os resultados corretos.

Deve-se ressaltar que nem sempre o período de *Burn-in* é necessário, havendo situações em que deve ser ignorado. A inclusão desse período deve ser avaliada pelo analista, uma vez que depende do problema que está sendo testado. Neste exemplo será utilizado um período de *Burn-in* de dois anos.

Para incluir esse período de *Burn-in* ao modelo, é necessário adicionar três novos blocos. Primeiramente, utiliza-se um bloco *Output Switch* para separar os pacientes que serão removidos. Isto é feito alterando nas configurações desse bloco o *Switching criterion* para a opção *From signal port p*, e associando a essa porta nova porta *p* um bloco *Step*, disponível na biblioteca *Simulink Sources*. Esse bloco cria um sinal com um determinado valor do início da simulação até um tempo predefinido, a partir do qual altera o sinal para um novo valor.

A Figura 43 mostra o modelo após a inclusão dos blocos para o *Burn-in*.

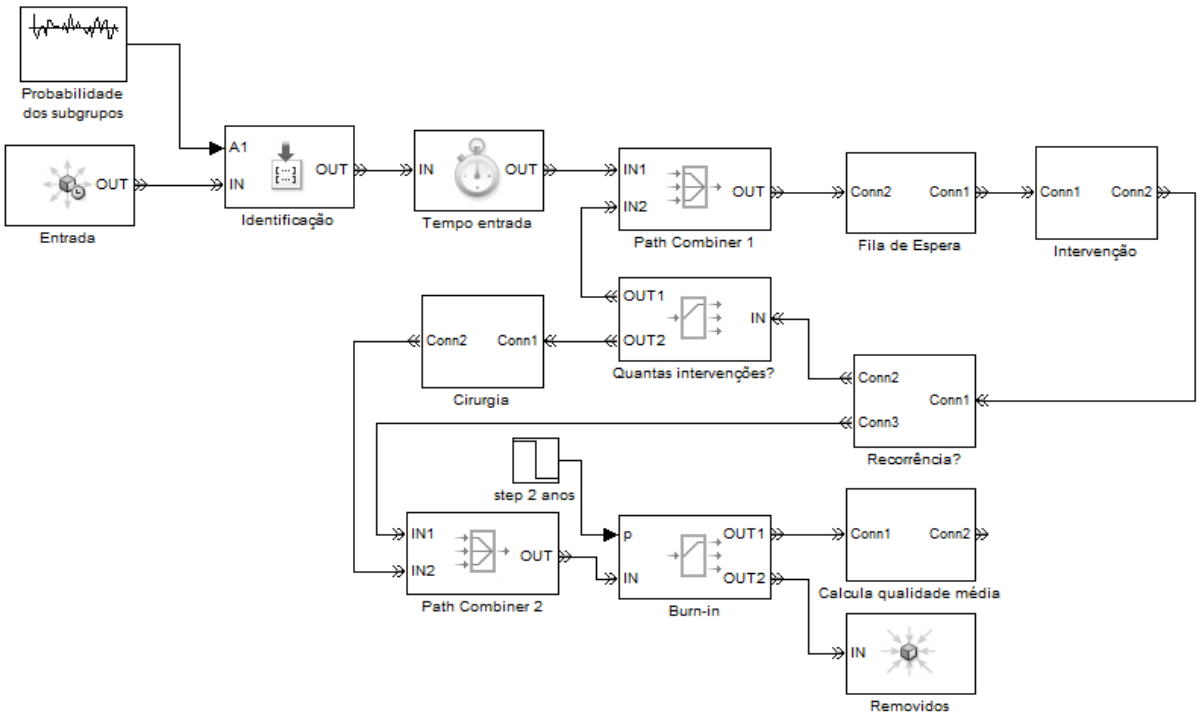

Figura 43 - Representação do modelo após a inclusão do *burn-in*.

O caminho dos pacientes pode ser controlado fazendo com que até certo tempo o sinal do bloco *Step* possua o valor 2, e ligando a porta 2 do bloco *Switch* a um bloco *Entity Sink*, disponível na biblioteca *SimEvents Sinks*, que remove os pacientes do modelo. Após esse tempo altera-se o valor para 1, fazendo com que os próximos pacientes sigam por um caminho diferente, onde serão observados o custo e a qualidade de vida média.

4.3.12. Obtendo os resultados

Uma maneira de se obter os resultados da simulação é utilizando o bloco *Discrete Event Signal to Workspace*, da biblioteca *SimEvents Sinks*. Este bloco salva os valores calculados por qualquer um dos blocos do modelo em uma variável na área de trabalho do MATLAB. Esta nova variável pode ser então utilizada para calcular medidas estatísticas, tais como média e desvio padrão.

A variável pode ser criada com diferentes formatos, como um *Array*, que armazena somente os valores observados, ou então como uma *Structure with time*, que contém os valores e o tempo em que cada valor foi observado. Neste exemplo é utilizada a opção *Array*.

O custo total e a qualidade de vida média de cada paciente podem ser obtidos através de um bloco *Get Attribute* configurado para ler os atributos "Custo" e "Qualidade".

Cada atributo precisa ser ligado a um bloco *Discrete Event Signal to Workspace* diferente, que deve ser configurado com o nome da variável que será criada e o formato utilizado. Por fim, depois que as medidas de interesse forem obtidas o bloco *Entity Sink* remove o paciente do modelo. A Figura 44 mostra o modelo completo.

Como diversos geradores de números aleatórios estão sendo utilizados nesse modelo, existe certa variabilidade nos resultados das médias que serão obtidas. Uma maneira de diminuir isso é através da realização de várias repetições da mesma simulação. Para isso, precisamos definir todas as sementes como um valor que se altere automaticamente a cada simulação, e também definir variáveis para os parâmetros que serão alterados ou testados. Após isso, é possível criar um *Script* contendo uma programação em MATLAB que realizará a execução das repetições. Este *Script* pode ser criado escolhendo a opção *File* → *New* → *Script* na janela inicial do MATLAB. O Apêndice B mostra um exemplo de código que realiza as repetições com o modelo que foi criado.

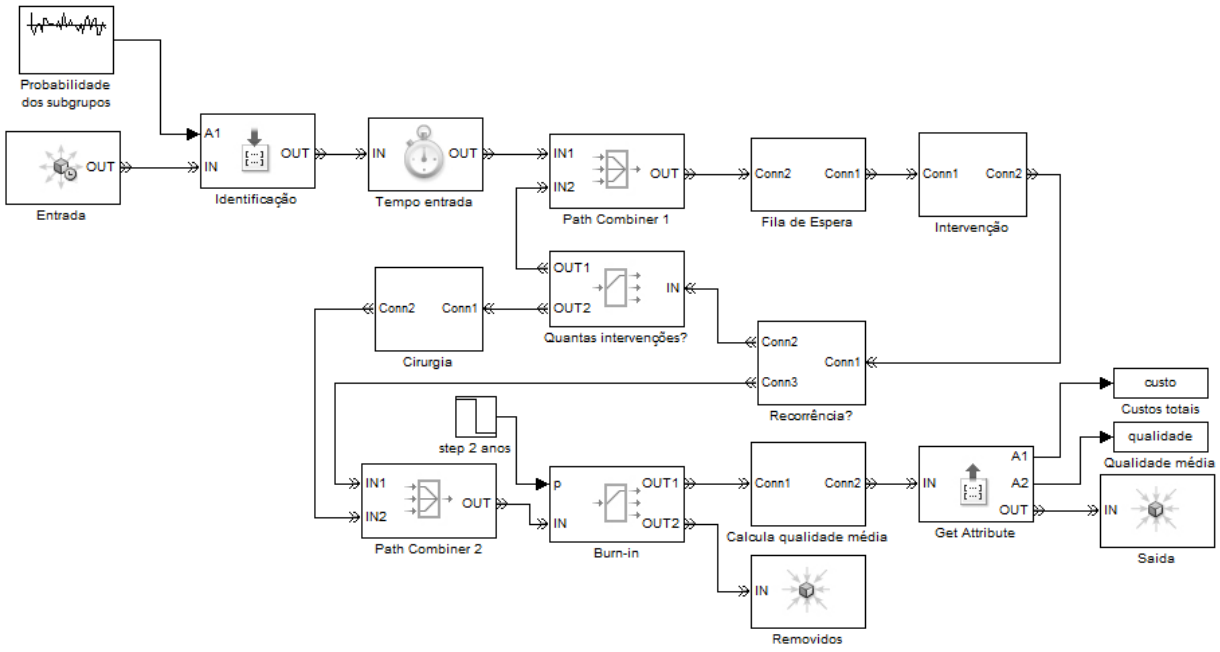

Figura 44 - Modelo completo.

Depois que as simulações forem executadas, os valores que salvos nas variáveis custo e qualidade podem ser acessados pela área de trabalho do MATLAB, onde os valores podem ser visualizados. É possível analisar os resultados no próprio MATLAB, ou então exportar os valores e realizar as análises em outro programa.

4.4. Resultados

O objetivo principal do problema de decisão relatado neste exemplo era comparar quatro possíveis estratégias para a aplicação de *stents* em pacientes que sofrem com a doença arterial coronariana.

Essas estratégias foram avaliadas com quatro diferentes limites no número máximo de atendimentos por dia por meio de um modelo de simulação de eventos discretos, pois esta é a única técnica capaz de trabalhar com filas de espera. Entretanto, quando esse limite é retirado pode-se utilizar um modelo de árvore de decisão para avaliar as quatro estratégias.

A Tabela 8 mostra os resultados que foram obtidos utilizando as duas técnicas quando o limite é desconsiderado. O custo e a qualidade de vida para a árvore de decisão representam os valores médios por paciente, conforme descritos anteriormente. Para a simulação de eventos discretos o custo reflete o valor médio gasto com cada paciente que foi tratado durante os sete anos avaliados pelo modelo, e a qualidade representa a qualidade de vida média desses pacientes durante o período que permaneceram recebendo o tratamento.

Tabela 8 - Comparação dos resultados obtidos com as duas técnicas.

		BBBB	BBDB	DBDB	DBDD
Árvore de decisões	Custo (€)	5269,67	5284,28	5446,32	5516,58
	Qualidade	0,84756	0,84760	0,84768	0,84768
Simulação de eventos discretos	Custo (€)	5268,14	5286,30	5446,14	5515,58
	Qualidade	0,84757	0,84761	0,84769	0,84769

Pode-se ver que os valores médios obtidos pelas duas técnicas são muito parecidos. Isso deverá acontecer sempre que não existir nenhum tipo de interação entre os indivíduos, já que é possível utilizar as duas técnicas para modelar essas situações, e os resultados obtidos serão equivalentes. Porém, quando existe uma interação entre os indivíduos o modelo de árvore de decisão torna-se impraticável e a simulação de eventos discretos se torna uma alternativa interessante.

Neste exemplo a fila de espera possui um efeito visível nos custos e na qualidade de vida. À medida que a capacidade de atendimentos por dia diminui, o tamanho da fila aumenta consideravelmente, levando a um aumento nos custos e a uma diminuição da qualidade de vida dos pacientes, como pode ser visto na Tabela 9.

Tabela 9 - Resultados de custo e efetividade para cada estratégia e capacidade de atendimento.

Limite		BBBB	BBDB	DBDB	DBDD
Sem limite	Custo (€)	5268,14	5286,30	5446,14	5515,58
	Qualidade	0,84757	0,84761	0,84769	0,84769
42	Custo (€)	5317,65	5300,02	5448,74	5517,21
	Qualidade	0,84622	0,84714	0,84761	0,84760
40	Custo (€)	5997,46	5882,14	5857,64	5926,31
	Qualidade	0,82973	0,83259	0,83696	0,83704
38	Custo (€)	6831,03	6750,14	6699,68	6772,26
	Qualidade	0,81419	0,81581	0,81943	0,81940

Quando o limite é desconsiderado a estratégia BBBB possui o menor custo dentre as quatro avaliadas, e a qualidade de vida média não é muito menor do que com as demais. Porém, conforme o limite diminui, essa estratégia passa a ser menos eficiente, passando a ser mais cara que as outras três quando é considerado um limite de 40 atendimentos por dia.

Fazendo a razão de custo-efetividade total para as três estratégias que incluem o uso dos novos *stents* em relação à estratégia BBBB, pode-se ver que quando o limite não é considerado a estratégia BBBB não é dominada por nenhuma outra, e o custo incremental é muito alto para qualquer uma das três novas estratégias. No entanto, se for considerado um limite de 42 atendimentos, a estratégia BBBB passa a ser dominada pela estratégia BBDB e o custo-incremental de se utilizar as outras duas estratégias passa a ser mais aceitável.

Assim, conforme a capacidade diminui, as novas estratégias vão sendo mais eficientes em relação à estratégia BBBB devido à menor taxa de recorrência destas, o que leva a uma menor utilização da fila de espera e, consequentemente, menor custo e melhor qualidade de vida para os pacientes. Os valores calculados para a razão de custo-efetividade incremental podem ser vistos na Tabela 10.

Tabela 10 - Razão de custo-efetividade incremental para as novas estratégias.

Limite	Razão de custo-efetividade incremental		
	BBBB - BBDB	BBBB - DBDB	BBBB - DBDD
Sem limite	404.833,89	1.442.810,02	1.969.461,66
42	Dominada	94.608,54	144.610,17
40	Dominada	Dominada	Dominada
38	Dominada	Dominada	Dominada

Se o limite no número de atendimentos for removido, é possível calcular o número médio de intervenções realizadas por dia. Isso pode ser útil para planejar qual a capacidade que pode ser requerida de modo a atender satisfatoriamente todos os pacientes. A Tabela 11 mostra o número médio de atendimentos por dia para cada estratégia. Como esperado, a utilização das novas *stents* revestidas com fármacos leva a um menor número de atendimentos, uma vez que diminui a taxa de recorrência dos pacientes.

Tabela 11 - Estimativa do número de pacientes atendidos por dia para cada estratégia.

	BBBB	BBDB	DBDB	DBDD
Atendimentos por dia	42,89	42,62	42,00	41,99
IC 95%	(42,80 ; 42,98)	(42,52 ; 42,72)	(41,92 ; 42,09)	(41,88 ; 42,10)

Esses resultados mostram que as novas *stents* podem proporcionar uma melhora no atendimento dos pacientes com doença arterial coronariana, pois diminuem a taxa de recorrência para esses pacientes, e que são mais eficientes principalmente quando existe o número de atendimentos que podem ser realizados por dia é limitado.

Outro ponto importante que deve ser verificado é o comportamento dos custos ao longo do tempo. Quando o número de atendimentos é limitado, surgem filas de espera que irão aumentar conforme o tempo, e se não for tomada nenhuma medida para corrigir o problema a situação continuará a piorar.

A Figura 45 mostra a evolução do número de pacientes na fila de espera ao longo do tempo para as quatro estratégias com cada limite de atendimentos por dia. Podemos ver que a fila aumenta rapidamente quando o limite é menor que o número de atendimentos necessários por dia, evidenciando uma situação insustentável, já que os pacientes precisaram aguardar cada vez mais tempo para serem atendidos.

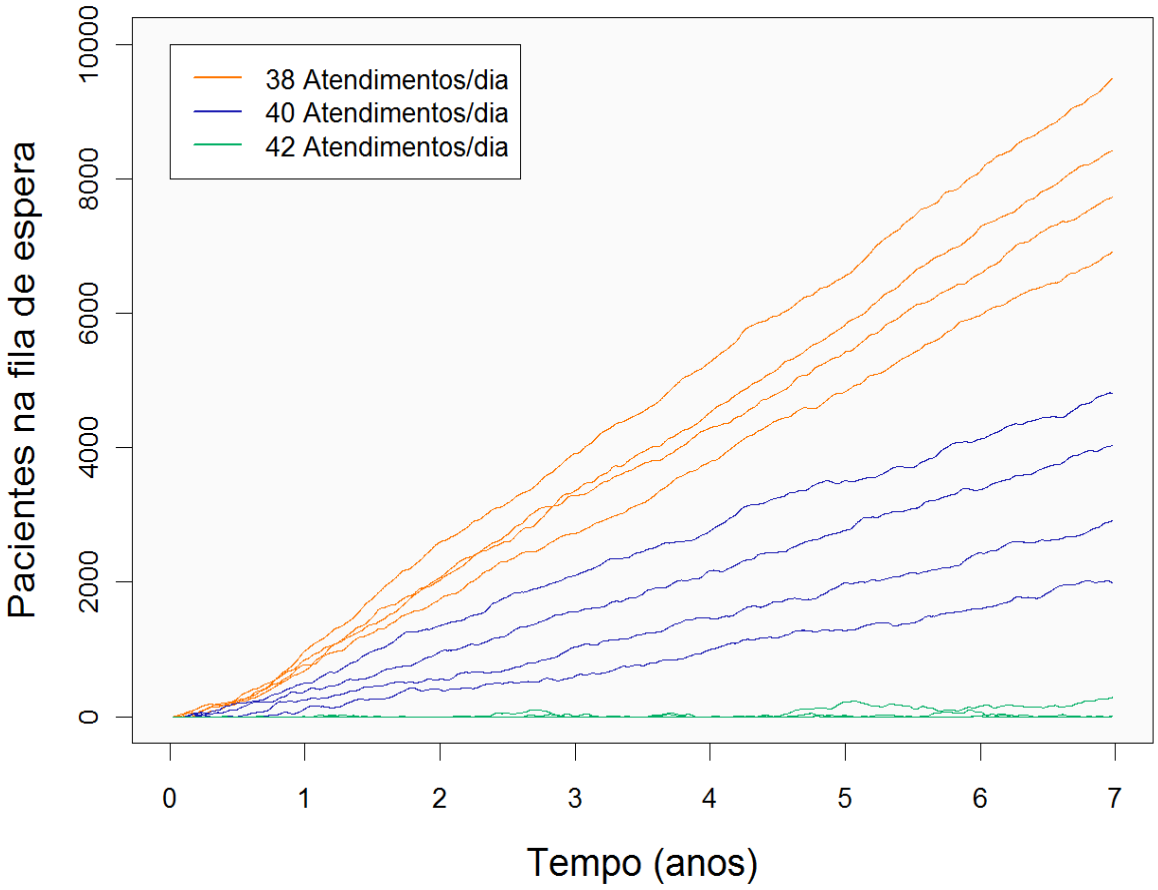

Figura 45 - Evolução do número de pacientes na fila de espera ao longo do tempo.

5. Considerações Finais

As árvores de decisão e os modelos de Markov são as duas técnicas mais amplamente utilizadas na avaliação de tecnologias em saúde. A simulação de eventos discretos, por sua vez, ainda é uma técnica pouco utilizada, mas que pode apresentar diversas vantagens em relação às outras, principalmente em situações que envolvam elevado grau de complexidade, além de ser extremamente flexível, sendo capaz de representar as mais variadas situações ou problemas de decisão.

Neste trabalho tentou-se descrever de maneira mais detalhada o conceito que envolve a simulação de eventos discretos e as formas de construir um modelo que utilize essa técnica. Ao mesmo tempo, procurou-se descrever de maneira mais simplificada os conceitos e a forma como são empregadas as árvores de decisão e os modelos de Markov na construção de um modelo, bem como seus principais usos e limitações.

Ainda, todas as técnicas de simulação que podem ser utilizadas apresentam algum tipo de vantagem ou de limitação em relação às outras, fazendo com que algumas sejam mais indicadas para representar certos tipos de situações. Sendo assim, nesse trabalho tentou-se também clarear as diferenças existentes entre as três técnicas descritas, com o intuito de auxiliar na escolha da técnica mais adequada de acordo com as características que são necessárias para o problema em questão.

Para auxiliar no entendimento dos tipos de modelagem apresentou-se um exemplo com o propósito de evidenciar as diferenças que podem ocorrer nos resultados de uma simulação por causa da escolha incorreta da técnica utilizada na construção do modelo. Através desse exemplo foi possível demonstrar a aplicabilidade da simulação de eventos discretos para avaliar situações que envolvem dependência ou interação entre pacientes, como em situações em que existe uma fila de espera para os pacientes receberem um atendimento.

Por fim, descreveu-se detalhadamente a construção de um modelo de eventos discretos com o objetivo de torná-lo facilmente reprodutível, na esperança de que este seja útil e que possa servir como base para quem desejar criar um modelo utilizando esta mesma técnica.

Em trabalhos futuros podem ser exploradas novas técnicas, como os sistemas dinâmicos, comparando as vantagens e desvantagens destas em relação àquelas que foram abordadas nesse trabalho e demonstrando situações em que elas poderiam ser aplicadas.

Referências Bibliográficas

BARTON, P.; BRYAN, S.; ROBINSON, S. Modelling in the economic evaluation of health care: selecting the appropriate approach. **Journal of Health Services Research & Policy,** v. 9, n. 2, p. 110-118, 2004. ISSN 13558196.

BRAILSFORD, S. C. System dynamics: What's in it for healthcare simulation modelers. Simulation Conference, 2008. WSC 2008. Winter, 2008. 7-10 Dec. 2008. p.1478-1483.

BRENNAN, A.; CHICK, S. E.; DAVIES, R. A taxonomy of model structures for economic evaluation of health technologies. **Health Economics,** v. 15, n. 12, p. 1295-1310, 2006. ISSN 1099-1050.

CARO, J. J.; MÖLLER, J.; GETSIOS, D. Discrete Event Simulation: The Preferred Technique for Health Economic Evaluations? **Value in Health,** v. 13, n. 8, p. 1056-1060, 2010. ISSN 1098-3015.

DRUMMOND, M. F. et al. **Methods for the Economic Evaluation of Health Care Programmes**. Oxford University Press, 2005. ISBN 0192627732.

FENDRICK, A. M. The Future of Health Economic Modeling: Have We Gone Too Far or Not Far Enough? **Value in Health,** v. 9, n. 3, p. 179-180, 2006. ISSN 1524-4733.

HAWKINS, N.; SCULPHER, M.; EPSTEIN, D. Cost-Effectiveness Analysis of Treatments for Chronic Disease: Using R to Incorporate Time Dependency of Treatment Response. **Medical Decision Making,** v. 25, n. 5, p. 511-519, September/October 2005.

JAHN, B. et al. Tutorial in Medical Decision Modeling Incorporating Waiting Lines and Queues Using Discrete Event Simulation. **Value in Health,** v. 13, n. 4, p. 501-506, 2010. ISSN 1098-3015.

KARNON, J.; BROWN, J. Selecting a decision model for economic evaluation: a case study and review. **Health Care Management Science,** v. 1, n. 2, p. 133-140, 1998. ISSN 1386-9620.

LE LAY, A. et al. Can discrete event simulation be of use in modelling major depression? **Cost Eff Resour Alloc,** v. 4, p. 19, 2006. ISSN 1478-7547.

MATHWORKS, I. **How do I install Microsoft Visual C++ 2010 Express and Microsoft Windows SDK 7.1?** 23 mar. 2011a. Disponível em: < http://www.mathworks.com/support/solutions/en/data/1-ECUGQX >. Acesso em: out. 2011.

MATHWORKS, I. **Why is 'mex -setup' unable to locate the Microsoft Platform SDK?** 19 nov. 2011b. Disponível em: < http://www.mathworks.com/support/solutions/en/data/1-425C91 >. Acesso em: out. 2011.

PHILIPS, Z. et al. Good Practice Guidelines for Decision-Analytic Modelling in Health Technology Assessment: A Review and Consolidation of Quality Assessment. **PharmacoEconomics,** v. 24, n. 4, p. 355, 2006. ISSN 11707690.

PIDD, M. **Computer Simulation in Management Science**. John Wiley & Sons, 2004. ISBN 0470092300.

STAHL, J. E. Modelling Methods for Pharmacoeconomics and Health Technology Assessment: An Overview and Guide. **PharmacoEconomics,** v. 26, n. 2, p. 131-148, 2008. ISSN 11707690.

VAN GESTEL, A. et al. Modeling Complex Treatment Strategies: Construction and Validation of a Discrete Event Simulation Model for Glaucoma. **Value in Health,** v. 13, n. 4, p. 358-367, 2010. ISSN 1098-3015.

WEINSTEIN, M. C. et al. Principles of Good Practice for Decision Analytic Modeling in Health-Care Evaluation: Report of the ISPOR Task Force on Good Research Practices—Modeling Studies. **Value in Health,** v. 6, n. 1, p. 9-17, 2003. ISSN 1524-4733.

WILLAN, A. R.; BRIGGS, A. H. **Statistical analysis of cost-effectiveness data**. Chichester, England: John Wiley & Sons, 2006. ISBN 0470856262.

Apêndice A – Função para o cálculo da qualidade de vida até um ano após a aplicação de *stents*

```
%--------------------------------------------------------------------------
% Define a função que calcula a qualidade de vida no período de um ano
%  após a aplicação da stent
% A função recebe os valores dos atributos TempoReoc e Qualidade
% A variável out_Qualidade faz com que os resultados sejam salvos novamente
%  no atributo Qualidade
%--------------------------------------------------------------------------
function out_Qualidade = fcn(TempoReoc, Qualidade)

%--------------------------------------------------------------------------
% Define quanto vale um mês e cria uma variável para a qualidade calculada
%--------------------------------------------------------------------------
mes = 365/12;
QualiReoc = 0;

%--------------------------------------------------------------------------
% Calcula a qualidade acumulada com o tempo, separando em intervalos
% A função interp1 retorna o valor de um ponto intermediário em uma reta
% A função trapz calcula a área sob uma curva formada por vários pontos
%--------------------------------------------------------------------------
if TempoReoc <= 1*mes
    QualiFim  = interp1([0,1*mes],[0.69,0.84],TempoReoc);
    QualiReoc = trapz  ([0,TempoReoc],[0.69,QualiFim]);
end
if (TempoReoc > 1*mes) && (TempoReoc <= 6*mes)
    QualiFim  = interp1([1*mes,6*mes],[0.84,0.86],TempoReoc);
    QualiReoc = trapz  ([0,1*mes,TempoReoc],[0.69,0.84,QualiFim]);
end
if (TempoReoc > 6*mes) && (TempoReoc <= 12*mes)
    QualiReoc = trapz  ([0,1*mes,6*mes,TempoReoc],[0.69,0.84,0.86,0.86]);
end
if (TempoReoc > 12*mes)
    QualiReoc = trapz  ([0,1*mes,6*mes,12*mes],[0.69,0.84,0.86,0.86]);
end

%--------------------------------------------------------------------------
% Soma ao valor do atributo Qualidade o valor que foi calculado
%--------------------------------------------------------------------------
out_Qualidade = Qualidade + QualiReoc;
```

Apêndice B – Código para executar várias simulações

```
%--------------------------------------------------------------------------
% Define o número de repetições para cada cenário
%--------------------------------------------------------------------------
nsim = 25;

%--------------------------------------------------------------------------
% Define o tempo médio até a recorrência, em dias;
%  Com lambda = taxa anual, o tempo médio é dado por 365/lambda;
%  O operador .^ eleva cada valor de um vetor à uma determinada potência.
%--------------------------------------------------------------------------
taxas.bare = 365 * [0.095 0.051 0.143 0.055].^(-1);
taxas.drug = 365 * [0.054 0.054 0.069 0.051].^(-1);

%--------------------------------------------------------------------------
% Define o número máximo de atendimentos por dia;
% A variável "limite" é associada ao bloco que controla a fila de espera;
% Os valores utilizados no exemplo foram [Inf, 42, 40, 38].
%--------------------------------------------------------------------------
limite = Inf;

%--------------------------------------------------------------------------
% Controla o custo da aplicação de um stent e o tempo de recorrência para
% cada subgrupo de acordo com a estratégia que está sendo testada;
% A variável "CustoSub" é associada ao bloco que controla o custo;
%  1 = stent convencional; 2 = stent com fármacos;
% A variável "ReocSub" contém o tempo de recorrência para cada subgrupo;
%  O gerador que controla o subgrupo x recebe o valor ReocSub(x).
%--------------------------------------------------------------------------
%  Estratégia BBBB  %
CustoSub = [1 1 1 1];
ReocSub = [taxas.bare(1) taxas.bare(2) taxas.bare(3) taxas.bare(4)];

%--------------------------------------------------------------------------
% Cria matrizes temporárias para os valores médios de custo e qualidade
%--------------------------------------------------------------------------
CustoTemp = zeros(nsim,1); QualiTemp = zeros(nsim,1);


%--------------------------------------------------------------------------
% Realiza as simulações e armazena os valores médios nas matrizes
%--------------------------------------------------------------------------
for i = 1:nsim
    sim('Stents_sim')
    CustoTemp (i) = mean(custo);
    QualiTemp (i) = mean(qualidade);
end


%--------------------------------------------------------------------------
% Organiza os valores obtidos durante a simulação
%--------------------------------------------------------------------------
CustoMedio.Inf.BBBB = CustoTemp; QualiMedio.Inf.BBBB = QualiTemp;
```

```
%-------------------------------------------------------------------------
% Realiza as simulações para as outras três estratégias
%-------------------------------------------------------------------------

%  Estratégia BBDB  %
CustoSub = [1 1 2 1];
ReocSub = [taxas.bare(1) taxas.bare(2) taxas.drug(3) taxas.bare(4)];
CustoTemp = zeros(nsim,1); QualiTemp = zeros(nsim,1);
for i = 1:nsim
    sim('Stents_sim')
    CustoTemp (i) = mean(custo);
    QualiTemp (i) = mean(qualidade);
end
CustoMedio.Inf.BBDB = CustoTemp; QualiMedio.Inf.BBDB = QualiTemp;

%  Estratégia DBDB  %
CustoSub = [2 1 2 1];
ReocSub = [taxas.drug(1) taxas.bare(2) taxas.drug(3) taxas.bare(4)];
CustoTemp = zeros(nsim,1); QualiTemp = zeros(nsim,1);
for i = 1:nsim
    sim('Stents_sim')
    CustoTemp (i) = mean(custo);
    QualiTemp (i) = mean(qualidade);
end
CustoMedio.Inf.DBDB = CustoTemp; QualiMedio.Inf.DBDB = QualiTemp;

%  Estratégia DBDD  %
CustoSub = [2 1 2 2];
ReocSub = [taxas.drug(1) taxas.bare(2) taxas.drug(3) taxas.drug(4)];
CustoTemp = zeros(nsim,1); QualiTemp = zeros(nsim,1);
for i = 1:nsim
    sim('Stents_sim')
    CustoTemp (i) = mean(custo);
    QualiTemp (i) = mean(qualidade);
end
CustoMedio.Inf.DBDD = CustoTemp; QualiMedio.Inf.DBDD = QualiTemp;
```