Pontifícia Universidade Católica de Minas Gerais

Programa de Pós-Graduação em Informática

# INFERÊNCIA DE IDENTIDADE GENÉTICA EM LARGA ESCALA: uma abordagem baseada em verificação simbólica de modelos

Carolina Maria Francisco Cota

Belo Horizonte

2009

**Carolina Maria Francisco Cota**

# INFERÊNCIA DE IDENTIDADE GENÉTICA EM LARGA ESCALA: uma abordagem baseada em verificação simbólica de modelos

Dissertação apresentada ao Programa de Pós-Graduação em Informática como requisito parcial para obtenção do Grau de Mestre em Informática pela Pontifícia Universidade Católica de Minas Gerais.

Orientador: Mark Alan Junho Song
Co-orientador: Rodrigo Richard Gomes

**Belo Horizonte**
**2009**

FICHA CATALOGRÁFICA

Elaborada pela Biblioteca da Pontifícia Universidade Católica de Minas Gerais

C843i

Cota, Carolina Maria Francisco
Inferência de identidade genética em larga escala: uma abordagem baseada em verificação simbólica de modelos / Carolina Maria Francisco Cota. – Belo Horizonte, 2009.
72f. : il.

Orientador: Mark Alan Junho Song.
Dissertação (Mestrado) – Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-graduação em Informática.
Bibliografia.

1. Bioinformática – Teses. 2. Genética legal. 3. Métodos formais (Computação). I. Song, Mark Alan Junho. II. Pontifícia Universidade Católica de Minas Gerais. III. Título.

CDU: 681.3.03:575.1

Bibliotecário: Fernando A. Dias – CRB6/1084

PUC Minas
Programa de Pós-graduação em Informática

# FOLHA DE APROVAÇÃO

*"Inferência de Identidade Genética – Uma Abordagem Baseada em verificação Simbólica de Modelos"*

**CAROLINA MARIA FRANCISCO COTA**

Dissertação defendida e aprovada pela seguinte banca examinadora:

Prof. Mark Alan Junho Song - Orientador (PUC Minas)
Doutor em Ciência da Computação - UFMG

Prof. Rodrigo Richard Gomes Co-orientador (PUC Minas)
Doutor em Bioinformática - UFMG

Prof. Sergio Vale Aguiar Campos - UFMG
Doutor em Ciência da Computação – CMU, Estado Unidos

Prof. Luis Enrique Zárate Gálvez - (PUC Minas)
Doutor em Engenharia Metalúrgica de Minas - UFMG

Belo Horizonte, 20 de agosto de 2009.

*Aos meus pais, ao meu irmão e ao Baldini.*

*Amo vocês!*

# AGRADECIMENTOS

Agradeço aos meus orientadores, Mark Alan e Rodrigo Richard, pela oportunidade de desenvolver este trabalho e pelo apoio nesta caminhada.

Ao Cristiano, por sua enorme colaboração.

Aos meus pais, por seu amor incondicional e por estarem sempre ao meu lado.

Ao meu irmão, pela amizade.

Ao meu marido, Baldini, que sempre me incentivou e apostou em minha capacidade.

A todos que não foram citados acima, porém que tiveram participação na elaboração deste trabalho.

# RESUMO

As técnicas de identificação de vítimas de acidentes tradicionalmente utilizadas incluem reconhecimento visual, comparação de impressão digital, análise de registros odontológicos e arcadas dentárias. Em grandes desastres a utilização dessas técnicas é, em muitos casos, inviável, pois requer a comparação de informações coletadas após a morte com informações obtidas antes da morte, que, muitas vezes, não estão disponíveis. Uma alternativa às técnicas tradicionais é a identificação das vítimas através de exames em DNA. A abordagem mais frequentemente utilizada, após o sequenciamento das amostras, é a construção de uma rede bayesiana para determinar a probabilidade de que a vítima seja membro de uma família. Apesar de preciso, esse método requer grande processamento computacional e rapidamente se torna impraticável, conforme o número de vítimas aumenta. Considerando que são gastos, em média, 1 minuto e 10 segundos para verificar se uma vítima é parte de uma família utilizando um sistema automatizado para criação da rede bayesiana, para 1.000 vítimas e famílias, meio milhão de redes devem ser criadas, o que corresponde a aproximadamente 14 meses de processamento de CPU. Neste trabalho, o uso das técnicas de Verificação Simbólica de Modelos é proposto para reduzir esse problema. O objetivo principal é diminuir a quantidade de redes bayesianas necessárias à identificação das vítimas, realizando um pré-processamento para indicar qual é a vítima mais provável para cada família. Com a abordagem proposta, para 1.000 vítimas e famílias, aproximadamente 105 mil redes devem ser criadas, o que representa uma redução próxima a 80% no número de redes bayesianas necessárias para a solução do problema. Desta forma, seriam necessarios aproximadamente 3 meses para identificar todas as vítimas.

Palavras-chave: Métodos Formais, Verificação Simbólica de Modelos, Identidade Genética, Inferência Genética em Larga Escala.

# ABSTRACT

The identification of accident victims traditionally uses techniques such as visual recognition, fingerprint or dental records comparison. In case of mass disasters, when the number of victims reaches hundreds or even thousands, the use of these techniques becomes unfeasible, because requires the comparison between after death and before death information, which is often not available. An alternative to these traditional methods is DNA identification. The most frequently used approach, after samples are sequenced, is to construct a bayesian network to determine the likelihood that the victim is indeed part of a family. While precise, this process is computer intensive and quickly becomes too expensive to perform as the number of victims grows. Determining the relationship between a single victim and its family using a bayesian network takes about 1 minute and 10 seconds using an automated expert system. For 1,000 victims and families, about half a million networks would have to be computed, adding up to about 14 months of CPU time. In this paper the use of symbolic model checking techniques is proposed in order to alleviate this problem. The study aims to reduce the number of bayesian network computed, using symbolic model checking to indicate the most likely victim for each family. With the proposed approach, for 1,000 victims and families, around 105 thousand networks would have to be computed, representing a decrease of 80% in the number of Bayesian networks needed to solve the problem. By this way, around 3 months of CPU time would be necessary to identify all the victims.



Key-words: Formal Methods, Symbolic Model Checking, Genetic fingerprinting, Large Scale Genetic Identity Inference.

# LISTA DE FIGURAS

# LISTA DE TABELAS

**SUMÁRIO**

# 1 INTRODUÇÃO

Desastres aéreos, catástrofes naturais, guerras e, mais recentemente, ataques terroristas são alguns dos cenários que podem resultar em centenas ou até milhares de mortes. Tem-se, nesses casos, um grande desafio: a identificação das vítimas.

As técnicas de identificação de vítimas tradicionalmente utilizadas incluem: reconhecimento visual, comparação de impressão digital, análise de registros odontológicos e arcadas dentárias. Em grandes desastres, a utilização dessas técnicas é dificultada, pois requer a comparação de informações coletadas após a morte com informações obtidas antes da morte, que muitas vezes não estão disponíveis. Outra dificuldade diz respeito às condições em que os corpos das vítimas são encontrados. Por exemplo, no ataque às torres gêmeas do *World Trade Center* em 11 de setembro de 2001, em que o número estimado de mortos foi de 2.819 pessoas, foram encontrados apenas 289 corpos intactos e cerca de 20.000 fragmentos de corpos humanos (NYMAG, 2002). Situações como essa, exigem a utilização de outras técnicas para identificação de vítimas. A identificação pelo teste em DNA é uma delas.

Existem duas abordagens para identificação de vítimas por teste em DNA: comparação direta e comparação familiar (LAU; TAN; TAN, 2005). Na comparação direta, caso exista algum pertence da vítima como escova de cabelo ou escova de dentes, por exemplo, de onde seja possível extrair uma amostra de DNA, essa amostra pode ser utilizada para realizar uma comparação direta com a amostra coletada no local do desastre. Uma correspondência exata serve como identificação da vítima. Essa abordagem apresenta os mesmos problemas das técnicas tradicionais citadas anteriormente, já que também requer a comparação de informações coletadas após a morte com informações obtidas antes da morte da pessoa.

Já na abordagem por comparação familiar, caso a vítima possua familiares próximos, vivos e que possam doar amostras de DNA, essas amostras podem ser utilizadas para se calcular a probabilidade de parentesco.

Utilizando a segunda abordagem, o problema principal é confrontar os dados de cada vítima com cada uma das famílias a fim de identificar com que família cada uma das vítimas possui compatibilidade segundo as leis de genética mendeliana (ALBERTS et al., 2004). Esse problema requer grande processamento computacional, visto que com 1.000 vítimas, no pior caso, o perfil genético da primeira vítima deveria ser confrontado com os perfis genéticos dos membros de cada uma das 1.000 famílias, o da segunda vítima comparado aos das 999 famílias restantes, e assim sucessivamente. Torna-se imprescindível, dessa forma, o uso de métodos e ferramentas computacionais eficientes, confiáveis e de comprovada acurácia que identifiquem o indivíduo rapidamente.

A técnica mais frequentemente utilizada, após o sequenciamento das amostras de DNA, é a construção de uma rede bayesiana para determinar a probabilidade de que a vítima seja membro de uma determinada família. A utilização de redes bayesianas com esse objetivo já foi estudada anteriormente e se apresenta como uma técnica eficiente mesmo em casos complexos, onde a informação genética não está disponível para todas as pessoas (DAWID; MORTERA; VICARD, 2006).

Apesar de preciso, no cenário de identificação de vítimas em desastres de larga escala, a inferência de parentesco através de redes bayesianas é um processo que demanda grande processamento computacional, visto que para cada par família/vítima é necessário criar uma rede bayesiana para inferência das relações de parentesco. Considerando que são gastos, em média, 1 minuto e 10 segundos para verificar se uma vítima é parte de uma família utilizando um sistema automatizado para criação da rede bayesiana (GOMES, 2008), para 1.000 vitimas e famílias, aproximadamente meio milhão de redes devem ser criadas, o que corresponde a aproximadamente 14 meses de processamento de CPU.

Neste trabalho, o uso das técnicas de Verificação Simbólica de Modelos é proposto para auxiliar na identificação das vítimas de desastres de larga escala. O diagrama apresentado na Figura 1 ilustra o processo de identificação de vítimas com o sistema GENESys, desenvolvido neste trabalho.

O GENESys irá atuar realizando um pré-processamento no perfil genético das vítimas e de seus familiares. Utilizando as técnicas de Verificação Simbólica de Modelos, o GENESys conseguirá apontar quais vítimas possuem maior compatibilidade genética com cada família. O resultado deste pré-processamento é apresentado em um arquivo de resultados que será utilizado para confirmar a identidade das vítimas com a criação de redes bayesianas. Desta forma, não será mais necessário criar redes bayesianas para todos os pares de vítima/família do desastre. Serão criadas redes bayesianas apenas para os

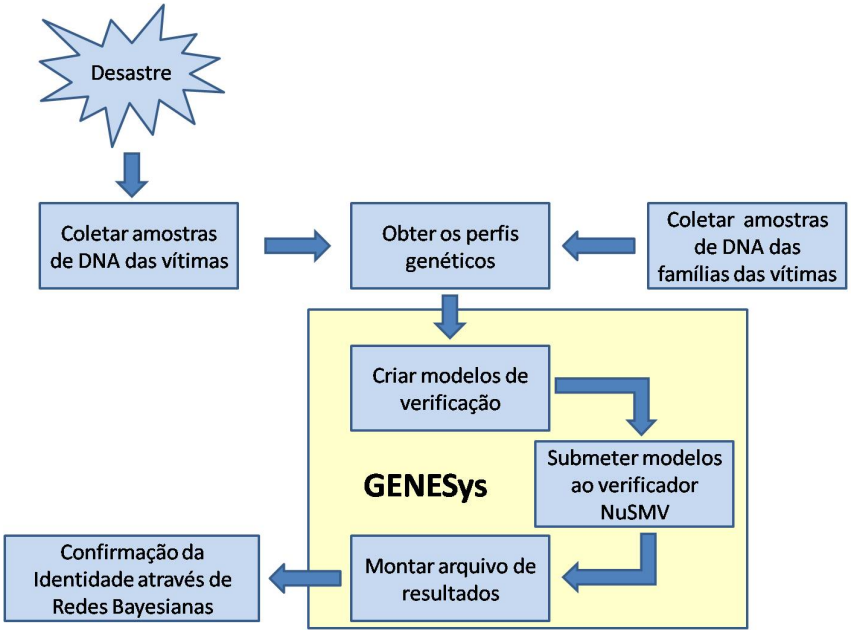



Figura 1: Diagrama ilustrativo do processo de identificação de vítimas com o sistema GENESys.

pares de vítima/família apontados no arquivo de resultados do GENESys.

Por exemplo, para 1.000 vitimas e famílias, após a execução do GENESys aproximadamente 105 mil redes Bayesianas devem ser criadas para confirmar a identidade das vítimas. Isso corresponde a uma redução próxima a 80% no número de redes e seriam necessários aproximadamente 3 meses de processamento de CPU.

## 1.1 Objetivos

Este trabalho tem como objetivo principal auxiliar na identificação de vítimas de desastres de larga escala, utilizando as técnicas de verificação simbólica de modelos para reduzir o número de redes bayesianas necessárias para identificar todas as vítimas do desastre.

Foram alcançados os seguintes objetivos específicos:

- Propor um modelo lógico para a verificação formal de relações de parentesco segundo as leis da genética mendeliana;
- Implementar um sistema computacional que:
  - Gere automaticamente os modelos formais baseado no modelo lógico proposto;
  - Execute os modelos formais no verificador NuSMV;

  - Interprete os resultados da verificação e gere um arquivo com resultados consolidados;

- Realizar simulações para validar a solução proposta.
- Apresentar os ganhos obtidos com a solução proposta.

## 1.2 Organização da Dissertação

Esta dissertação está assim organizada: esse capítulo a introduz, apresentando as motivações e objetivos. O Capítulo 2 apresenta os conceitos básicos e descreve as técnicas para inferência genética por DNA. O Capítulo 3 descreve as técnicas de Verificação Simbólica de Modelos, bem como o verificador NuSMV. O Capítulo 4 apresenta a solução proposta. Já o Capítulo 5 exibe os resultados obtidos em simulações realizadas, e finalmente o Capítulo 6 conclui esta dissertação.

# 2 IDENTIFICAÇÃO GENÉTICA

Nos últimos anos, com os avanços alcançados pela biologia molecular foi definitivamente comprovado que, com exceção dos gêmeos univitelinos, todos os seres humanos diferem em sua constituição genética, que é absolutamente única (PENA; PRADO; EPPLEN, 1995). Essa individualidade pode ser aplicada, entre outras coisas, na identificação de criminosos e vítimas.

Para compreender o funcionamento do processo de identificação genética, é necessário conhecer alguns conceitos básicos que serão descritos nas seções seguintes.

## 2.1 Fundamentos de Genética

Todo ser vivo é constituído por unidades fundamentais: as células. A maioria dos organismos vivos são unicelulares. Outros organismos, como os seres humanos, são formados por vastas estruturas multicelulares, nas quais grupos de células realizam funções especializadas e estão ligados por intrincados sistemas de comunicação.

Em ambos os casos, os organismos foram gerados a partir da divisão de uma única célula. Consequentemente, uma simples célula é o veículo de informação hereditária que define as espécies (ALBERTS et al., 2004). Tudo isso é possível porque as células contêm em seu núcleo informação genética que está armazenada em moléculas de DNA (*Deoxyribonucleic Acid*).

O conjunto completo de material genético de um organismo é chamado de genoma. O genoma humano contém em torno de 25.000 genes, que é a unidade base da hereditariedade (NUSSBAUM; MCINNES; WILLARD, 2002). Segundo Snustad e Simmons (2001), o genoma desempenha três funções essenciais:

1. A replicação (função genotípica) - o material genético deve armazenar as informações genéticas e transmití-las com precisão de pais para filhos, geração após geração;

2. A expressão gênica (função fenotípica) - o material genético deve controlar o desenvolvimento do organismo, ou seja, deve ditar o seu crescimento e diferenciação desde o nascimento até a morte;

3. A mutação (função evolutiva) - o material genético deve sofrer mudanças para que os organismos possam se adaptar a modificações no ambiente. Sem tais mudanças, a evolução não poderia ocorrer.

Para compreender como o material genético desempenha essas funções é importante conhecer a estrutura da molécula de DNA. O modelo de Watson e Crick (1953) revolucionou o pensamento da biologia moderna. Segundo esse modelo, o DNA é formado por uma sequência de nucleotídeos que estão organizados em duas fitas antiparalelas em forma de dupla hélice.

Os nucleotídeos são moléculas formadas por três partes: um açúcar de cinco carbonos (pentose), um grupo fosfato e uma base nitrogenada. No DNA, o açúcar é a desoxirribose e as bases nitrogenadas podem ser de dois tipos: purinas (Adenina [A] e Guanina [G]) e pirimidinas (Citosina [C] e Timina [T]). Os nucleotídeos são ligados uns aos outros pelas moléculas de fosfato-pentose, formando uma sequência repetitiva dessas moléculas.

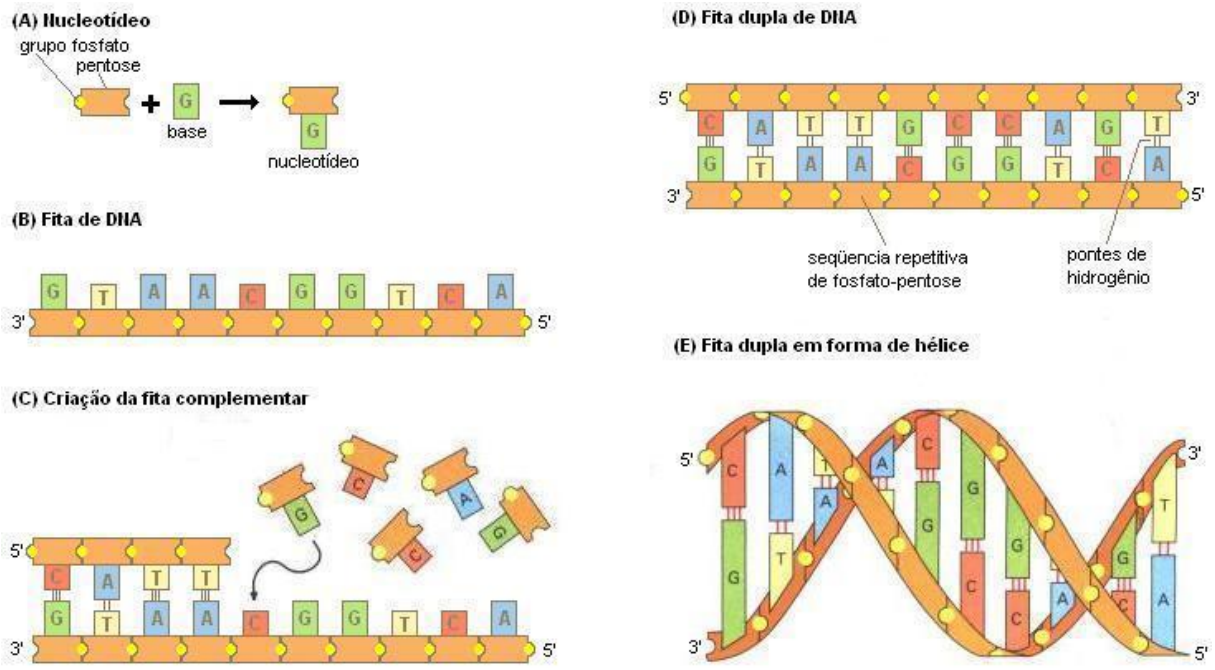



Figura 2: O Modelo de Watson-Crick da Estrutura do DNA

Todas as bases estão do lado interno da hélice e se ligam umas as outras por pontes de hidrogênio. Entretanto, as bases não se ligam ao acaso: Adenina [A] sempre se liga

com Timina [T] e Guanina [G] sempre se liga com Citosina [C]. São os chamados pares de base. A Figura 2 ilustra o modelo de Watson-Crick (ALBERTS et al., 2004).

Uma vez conhecida a sequência de bases de uma das fitas de DNA, a da outra é também conhecida, devido ao pareamento específico das bases. As duas fitas de uma dupla hélice de DNA são chamadas de complementares.

As ligações entre os pares de base são fracas, quando comparadas com as ligações entre as moléculas de fosfato-pentose do nucleotídeo, permitindo que as duas fitas de DNA sejam separadas sem que ocorram danos. Sendo assim, cada fita pode servir de molde para a síntese de uma nova fita de DNA complementar a si própria - isto é, uma nova cópia da informação hereditária. Em diferentes tipos de células, o processo de replicação de DNA ocorre em diferentes velocidades, com diferentes controles para iniciá-lo ou interrompê-lo (ALBERTS et al., 2004).

Em geral, a replicação do DNA entre células dá-se de maneira correta, mas, eventualmente, podem ocorrer erros, o que se denomina mutações (BORGES-OSóRIO; ROBINSON, 2001). Essas podem envolver a substituição, deleção ou inserção de base, conforme exemplificado na Figura 3.

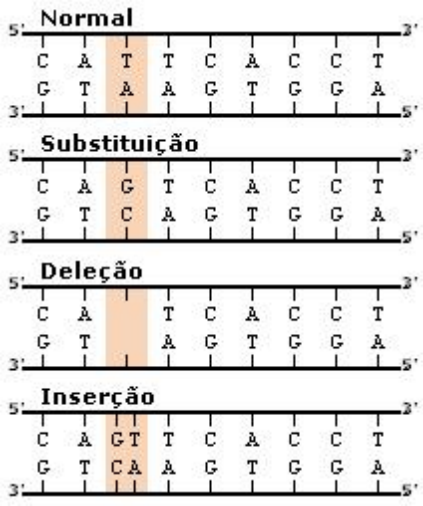



Figura 3: Exemplos de mutações gênicas

O DNA está dividido em uma série de diferentes cromossomos. Por exemplo, o genoma humano, de aproximadamente $3,2 \times 10^9$ nucleotídeos, está distribuído entre 24 diferentes cromossomos. Com exceção das células reprodutivas, cada célula humana contêm duas cópias de cada cromossomo, uma herdada da mãe e outra do pai. Os cromossomos maternos e paternos de um par são chamados cromossomos homólogos. O único par de cromossomos não-homólogos é o dos cromossomos sexuais do indivíduo masculino,

onde um cromossomo $Y$ é herdado do pai e o cromossomo $X$ é herdado da mãe (ALBERTS et al., 2004).

Assim, cada célula humana contém um total de 46 cromossomos, 22 pares estão presentes em homens e mulheres e são chamados de autossomos. Eles estão numerados em ordem decrescente do maior (cromossomo 1) até o menor (cromossomos 21 e 22). O par restante constitui os cromossomos sexuais: $XY$ nos homens e $XX$ nas mulheres (NUSSBAUM; MCINNES; WILLARD, 2002).

Cada cromossomo possui um subgrupo diferente de genes que estão dispostos linearmente ao longo de seu DNA, cada gene tendo uma posição exata ou loco (plural locos). Assim, cada par de cromossomo homólogo possui informações genéticas similares, isto é, têm os mesmos genes, na mesma sequência. Em um loco específico, entretanto, o gene pode apresentar diferentes formas. Cada uma das formas alternativas de um gene, ou de um marcador genético, é chamada alelo. Ou seja, qualquer mudança em um dos nucleotídeos de um gene cria um alelo diferente. A Figura 4 ilustra essa estrutura.

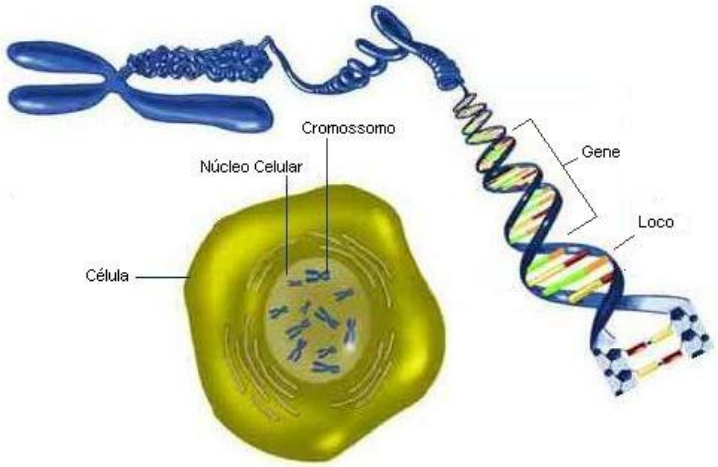



Figura 4: Estrutura Molecular

## 2.2 Heredogramas

Heredogramas são diagramas que mostram o parentesco entre os membros de uma família. É padrão representar os homens por quadrados e as mulheres por círculos. Uma linha horizontal unindo um quadrado a um círculo representa uma reprodução. Os filhos são mostrados logo abaixo dos pais, começando com o que nasceu primeiro à esquerda e continuando por ordem de nascimento para a direita. As gerações são geralmente indicadas por algarismos romanos, e as pessoas de uma geração por algarismos arábicos.

Pessoas falecidas são representadas com um corte em seu quadrado ou círculo (SNUSTAD; SIMMONS, 2001). Um conjunto de símbolos utilizados na criação de heredogramas e um exemplo é apresentado na Figura 5.

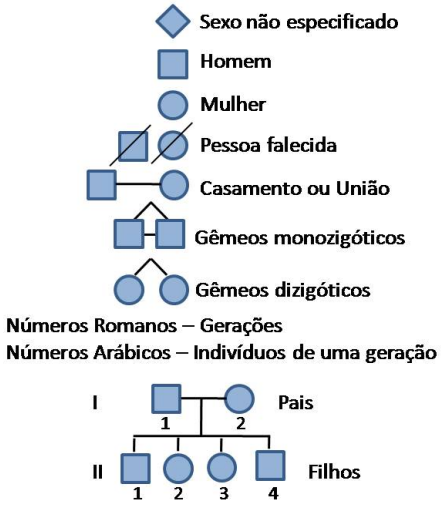



Figura 5: Convenções de heredograma com exemplo.

## 2.3 Perfil genético

As impressões digitais tiveram um papel fundamental na identificação de humanos durante décadas. O seu uso em casos forenses é baseado na premissa de que duas pessoas não tem impressões digitais iguais. Do mesmo modo, não existem duas pessoas, exceto gêmeos univitelinos, que tenham genomas com a mesma sequência de nucleotídeos (SNUSTAD; SIMMONS, 2001). Desta forma, pode-se utilizar o perfil genético para identificação de humanos.

Segundo Pena (2005), a determinação da identidade genética pelo exame em DNA pode ser considerada um dos produtos mais revolucionários da genética molecular humana moderna.

O perfil genético de uma pessoa consiste em medições de marcadores genéticos. Embora vários tipos de marcadores genéticos estejam disponíveis, *Short Tandem Repeats* (STR) é o tipo mais frequentemente utilizado para exames em DNA.

A maior parte do DNA humano é idêntico para todas as pessoas, entretanto, existem algumas regiões do DNA que variam de pessoa para pessoa. Essas variações são chamados polimorfismos: minissatélites, microssatélites ou STRs (*Short Tandem Repeats*).

Para extrair os polimorfismos, inicialmente utilizava-se sondas de DNA, que eram

capazes de reconhecer simultaneamente diversas regiões de minissatélites. Na segunda metade da década de 80 surgiu um método poderoso de amplificação molecular, chamado PCR (*Polymerase Chain Reaction*), que tornou possível o estudo de microssatélites ou STRs. Essa técnica é mais simples que a anterior e a sua utilização tornou o processo mais rápido e mais sensível, permitindo que o exame seja realizado com quantidades mínimas de DNA, mesmo em níveis avançados de degradação (PENA, 2005).

Nos locos de microsatélites, um conjunto, geralmente variando entre 2-5 pares de base, são continuamente e sequencialmente repetidos. O número de repetições contidas dentro de um loco pode diferir entre os dois cromossomos homólogos de uma pessoa e entre pessoas na população. Os diferentes números de repetições em um loco constituem os alelos deste loco. A Figura 6 apresenta um exemplo de repetições de AATG.

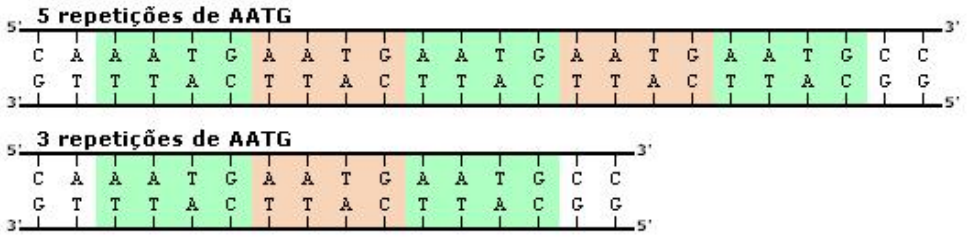



Figura 6: Exemplo de repetições de STR. A primeira sequência representa o alelo 5, e a segunda o alelo 3.

Ainda hoje, o PCR é uma das técnicas mais utilizadas nos exames em DNA e é nela que se baseiam os principais bancos de dados genéticos existentes pelo mundo. O primeiro banco de dados genético foi criado pelo Reino Unido em 1995 e em seguida foram criados outros na Nova Zelândia, países europeus, Estados Unidos e Canadá (WALSH, 2004). O mais famoso dos bancos de dados genéticos é o do FBI, chamado CODIS (*Combined DNA Index System*) (BUDOWLE; BIEBER; EISENBERG, 2005), que utiliza um conjunto de 13 locos de STR para determinar o perfil genético de uma pessoa.

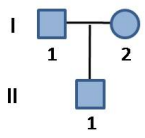
Além dos minissatélites e microssatélites existem outros dois grandes grupos de polimorfismos no genoma humano: os SNPs, que são polimorfismos com base na substituição de nucleotídeos únicos; e os Indels, que são polimorfismos de inserção ou deleção de um ou mais nucleotídeos. Esses dois tipos de polimorfismos possuem vantagens sobre os STRs no estudo de DNA extremamente degradado. Existem pesquisas sobre a possibilidade de que SNPs e Indels possam vir a substituir os STRs nos bancos de dados genéticos. Na opinião de Pena (2005), essa substituição está longe de acontecer, pois os bancos de dados já estão estruturados para trabalhar com STRs e as técnicas de extração de SNPs e Indels são mais complexas que o uso de PCR.

## 2.4 Princípios Mendelianos da Genética

Para interpretar os perfis genéticos obtidos e determinar se existe ou não uma relação de parentesco entre os indivíduos, utiliza-se os princípios mendelianos da genética.

Por ser um organismo diplóide e, portanto, possuir duas cópias de cada um dos 22 autossomos (os cromossomos não sexuais) além de 2 cromossomos sexuais (X e Y), o homem possui dois alelos para cada loco de polimorfismo, um herdado de seu pai e outro herdado de sua mãe. Caso esses alelos sejam iguais, são chamados homozigotos e caso sejam diferentes, heterozigotos.

Um exemplo de como os princípios Mendelianos são utilizados para determinar uma relação de parenteso é apresentado na Figura 7, que mostra um heredograma simples de um pai, mãe e filho, e uma tabela com os valores dos alelos para o loco de nome $VWA$. Pai e mãe possuem os alelos (16, 17) para esse loco. Assim, o filho poderá ser (16, 16), caso herde o valor 16 da mãe e outro valor 16 do pai; ou ser (16, 17), caso herde o valor 16 da mãe e o valor 17 do pai ou vice-versa; e também poderá ser (17, 17), caso herde o valor 17 da mãe e também o valor 17 do pai. Portanto, uma pessoa que tenha, por exemplo, o valor (15, 18) para este mesmo loco não poderá ser filho deste casal. Utilizando-se este mesmo raciocínio para vários locos de STR é possível calcular a probabilidade de que uma pessoa seja filha de um determinado casal.




| | VWA |
|---|---|
| I1 – Pai | 16,17 |
| I2 – Mãe | 16,17 |
| II1 - Filho | 16,16<br>16,17<br>17,17 |

Figura 7: Exemplo do principio mendeliano.

Dessa forma, quanto mais próximos forem duas pessoas em um heredograma, mais alelos elas têm em comum, herdados de seus ancestrais comuns. Ao contrário, quanto mais afastadas as duas pessoas tiverem no heredograma, menos alelos serão compartilhados entre elas.

Os exemplos mais extremos de duas pessoas tendo alelos em comum são os gêmeos idênticos (univitelinos), que têm os mesmos alelos para todos os locos. Em seguida, as pessoas relacionadas de maneira mais próxima em uma família são os parentes de primeiro grau, tais como um genitor e seu filho ou um par de irmãos. Em um par genitor-filho, a criança tem apenas um alelo em comum com cada genitor em cada loco, isto é, o alelo que

a criança herda deste genitor. Para um par de irmãos a situação é um pouco diferente. Um par de irmãos herda os mesmos dois alelos em um loco em 25% das vezes, nenhum alelo em comum em 25% das vezes e apenas um alelo em comum em 50% das vezes (NUSSBAUM; MCINNES; WILLARD, 2002).

## 2.5 Teorema de Bayes

O Teorema de Bayes é frequentemente utilizado para se determinar as probabilidades relativas de duas hipóteses mutuamente exclusivas (HOFFEE, 2000). Para cada hipótese, são calculados quatro componentes:

- *Probabilidade a priori* é a probabilidade inicial, sem considerar complicações ou informações adicionais.
- *Probabilidade condicional* é a probabilidade de uma informação adicional sob cada alternativa.
- *Probabilidade conjunta* é o produto das probabilidades *a priori* e condicional.
- *Probabilidade a posteriori* é a probabilidade conjunta de cada alternativa dividida pela soma das duas probabilidades conjuntas.

Considere o cenário: (PENA, 2006) uma mãe tem os alelos (14, 17) para um determinado loco de STR e seu filho possui os alelos (13, 17) para esse mesmo loco. Pode um indivíduo $F$ que possui os alelos (10, 13) para esse loco ser o pai da criança?

As hipóteses são: ou $F$ ou um outro indivíduo da população é o pai da criança, e essas hipóteses são mutuamente exclusivas, o que permite utilizar o teorema de bayes para solucionar a questão.

A probabilidade *a priori*, adotada por convenção internacional em casos de determinação de paternidade, é de $0,5$ (50%).

Pelo princípio mendeliano da genética sabemos que a criança herdou o alelo 17 de sua mãe e 13 de seu pai. $F$ possui um de seus alelos igual a 13, o que significa que a probabilidade que esse alelo 13 seja passado para um filho de $F$ é de $0,5$ (50%). Se $F$ tivesse os dois alelos iguais a 13 para esse loco a probabilidade seria de 100%. Já a probabilidade de que um indivíduo qualquer da população tenha o alelo 13 é determinada pela frequência do alelo 13 para esse loco na população em questão. Considerando que

essa probabilidade seja de 0.075 (7.5%), tem-se as probabilidades condicionais das duas hipóteses.

Agora é possível calcular a probabilidade *a posteriori* conforme apresentado na Tabela 1. A probabilidade de que $F$ seja o pai é de 87% contra 13% de probabilidade de que outro indivíduo da população seja o pai da criança.

| | F | Outro indivíduo |
|---|---|---|
| Probabilidade *a priori* | 0.5 | 0.5 |
| Probabilidade condicional | 0.5 | 0.075 |
| Probabilidade conjunta | $0.5 \times 0.5 = 0.25$ | $0.5 \times 0.075 = 0.0375$ |
| Normalização | $(0.25 + 0.0375 = 0.2875)$ | |
| Probabilidade *a posteriori* | $0.25/0.2875 = 0.87$ | $0.0375/0.2875 = 0.13$ |

Tabela 1: Resumo do cálculo pelo teorema de Bayes

Em um caso real de determinação de paternidade, esse cálculo é realizado para vários locos, o que permite alcançar um resultado preciso. Já que, dois genomas humanos escolhidos ao acaso diferem aproximadamente em 1 de cada 500 nucleotídeos. Como o genoma humano tem $3,2 \times 10^9$ nucleotídeos, isso implica em aproximadamente 6 milhões de diferenças (PENA; PRADO; EPPLEN, 1995), sendo praticamente impossível que duas pessoas sem nenhuma relação de parentesco tenham um grande número de STR locos com os mesmos valores.

As controvérsias quanto ao uso dos perfis genéticos em casos forenses estão relacionadas à competência dos laboratórios de pesquisa envolvidos, à probabilidade de erro humano na manipulação das amostras e testes, e aos métodos para calcular a probabilidade de que duas pessoas tenham o mesmo perfil genético (SNUSTAD; SIMMONS, 2001).

O Teorema de Bayes é um método probabilístico bastante utilizado com essa finalidade, mas para se fazer uma estimativa precisa da probabilidade de perfis genéticos idênticos, é necessário ter informações confiáveis sobre a frequência alélica dos polimorfismos na população em questão. No cenário apresentado acima, por exemplo, foi considerado que para o loco analisado a frequência do alelo 13 na população é de 0.075 (7.5%). Em uma outra população esse valor será outro e o resultado do cálculo também será diferente. Ou seja, os dados obtidos de uma população nunca devem ser extrapolados para outra, pois frequências diferentes de polimorfismo podem estar presentes em populações diferentes.

Como uma população inteira, em geral, é muito grande para ser estudada, o cálculo da frequência alélica é realizado analisando uma amostra representativa de indivíduos. Considere o exemplo apresentado por Snustad e Simmons (2001), onde a Tabela 2 apre-

senta dados de uma amostra de pessoas que foram testadas quanto aos tipos sanguíneo M-N. Estes tipos sanguíneos são determinados por dois alelos de um gene no cromossomo 4: o alelo $L^M$, que produz o tipo sanguíneo $M$, e $L^N$, que produz o tipo sanguíneo N. As pessoas que são heterozigotas $L^M L^N$ têm o tipo sanguíneo $MN$.

| Tipo Sanguíneo | Genótipo | Número de Pessoas |
|---|---|---|
| $M$ | $L^M L^M$ | 1.787 |
| $MN$ | $L^M L^N$ | 3.039 |
| $N$ | $L^N L^N$ | 1.303 |

Tabela 2: Frequência de tipos sanguíneos M-N em uma amostra de 6.129 pessoas

Para estimar as frequências dos alelos $L^M$ e $L^N$, simplesmente se calcula a incidência de cada alelo entre todos os alelos amostrados:

1. O número total de alelos na amostra é duas vezes o tamanho da amostra: $2 \times 6.129 = 12.258$.

2. A frequência do alelo $L^M$ é duas vezes o número de homozigotos $L^M L^M$ mais o número de heterozigotos $L^M L^N$, e o total dividido pelo número total de alelos amostrados: $[(2 \times 1.787) + 3.039]/12.258 = 0,5395$.

3. A frequência do alelo $L^N$ é duas vezes o número de homozigotos $L^N L^N$ mais o número de heterozigotos $L^M L^N$, e o total dividido pelo número total de alelos amostrados: $[(2 \times 1.303) + 3.039]/12.258 = 0,4605$.

Assim, se $p$ representar a frequência do alelo $L^M$ e $q$ representar a frequência do alelo $L^N$, avalia-se que na população da qual a amostra foi obtida $p = 0,5395$ e $q = 0,4605$. Além disso, como $L^M$ e $L^N$ são os únicos dois alelos deste gene em particular, $p + q = 1$.

# 3 MÉTODOS FORMAIS

Os sistemas computacionais são cada vez mais complexos e mais susceptíveis a erros. Construir sistemas confiáveis é um dos maiores desafios na engenharia de software. Uma forma de se minimizar as falhas é utilizar métodos formais.

Recentemente métodos formais e mais especificamente verificação simbólica de modelos têm sido utilizados para verificar propriedades de sistemas estocásticos como a verificação de redes bioquímicas (CHABRIER; FAGES, 2003) e o estudo de estruturas de proteínas (LANGMEAD; JHA, 2007).

Este trabalho propõe uma nova abordagem para a solução do problema de inferência de identidade genética com DNA baseando-se na verificação simbólica de modelos. Este capítulo apresenta uma introdução às técnicas de métodos formais e sua utilização.

## 3.1 Introdução

Métodos formais são linguagens, técnicas e ferramentas matemáticas usadas para especificar e verificar sistemas (CLARKE; WING; AL, 1996). O uso de métodos formais por si só não garante a qualidade do sistema, mas aumenta consideravelmente seu entendimento, revelando inconsistências, ambiguidades e imperfeições. Os métodos formais podem ser divididos em técnicas de especificação e técnicas de verificação.

As técnicas de especificação são utilizadas para formalizar os requisitos e propriedades de um sistema. A especificação de um sistema é uma ferramenta importante de comunicação entre os usuários do sistema e os projetistas, entre os projetistas e os desenvolvedores e entre os desenvolvedores e os testadores. É um documento importante para um bom entendimento do sistema. Alguns exemplos dessas técnicas são: Z (SPIVEY, 1988) e VDM (LARSEN; LASSEN, 1994).

Já as técnicas de verificação vão um passo além. Elas auxiliam os projetistas na identificação de possíveis erros. O sistema é modelado em uma linguagem adequada e

suas propriedades são formalmente descritas e verificadas. Duas abordagens comuns são os provadores de teoremas e a verificação de modelos.

Os provadores de teoremas usam lógica matemática para expressar um sistema e suas propriedades. O sistema é descrito por um conjunto $\Gamma$ de fórmulas e uma propriedade sobre o sistema também é uma fórmula $\phi$. O procedimento de verificação consiste em encontrar uma prova para $\Gamma \vdash \phi$. A verificação por provadores de teoremas é semi-automática e exige um especialista em lógica para conduzir a prova. É a técnica ideal para verificar sistemas que são modelados por um número infinito de estados.

Por sua vez, a verificação de modelos foi idealizada para sistemas concorrentes, que podem ser modelados por um conjunto finito de estados. Nesse aspecto, a verificação de modelos é significativamente mais simples que os provadores de teoremas. Por outro lado, a verificação de modelos é um procedimento mais rápido, automatizado e que apresenta um contra-exemplo quando a propriedade não é valida.

O maior desafio técnico da verificação simbólica de modelos tem sido o problema de explosão de estados. Esse problema ocorre em sistemas com muitos componentes que podem interagir entre si ou em sistemas que têm estruturas de dados que podem assumir muitos valores. Nesses casos, o número de estados alcançáveis no sistema pode ser enorme, inviabilizando a verificação (CLARKE; GRUMBERG; PELED, 1999; CLARKE et al., 2001).

A seção seguinte apresenta detalhes sobre as técnicas de verificação de modelos.

## 3.2 Verificação de modelos

Verificação de modelos consiste na representação de um sistema por meio de um modelo finito a ser exaustivamente analisado para determinar sua conformidade em relação à certas propriedades (CLARKE; GRUMBERG; PELED, 1999).

O sistema a ser verificado é representado como um grafo de transição de estados (o modelo) e as propriedades são descritas como fórmulas em uma linguagem temporal. No grafo, cada vértice corresponde a um estado do sistema, que é determinado pelos valores contidos em cada uma de suas variáveis. Já as arestas correspondem a transições entre os estados. O procedimento de verificação consiste em percorrer todos os estados do modelo e verificar se o mesmo atende as propriedades definidas.

Dessa forma, para realizar a validação das propriedades deve-se seguir os três passos abaixo:

1. Especificar quais são as propriedades que o sistema deverá ter para ser considerado correto. Por exemplo, pode-se querer que o sistema nunca entre em *deadlock*, ou ainda, que ele sempre alcance um determinado estado.

2. O segundo passo é a construção de um modelo formal para representar o sistema. O modelo deve capturar todas as propriedades essenciais à verificação, abstraindo detalhes que não afetem a correção dessas propriedades. Por exemplo, em protocolos de comunicações se está interessado em testar o recebimento de uma mensagem, e não no conteúdo dessa mensagem.

3. O terceiro e último passo é a própria execução do verificador de modelos para validar as propriedades especificadas no primeiro passo sobre o modelo criado no segundo passo. Desta forma, aplica-se o verificador para testar se o modelo atende as propriedades desejadas. Caso todas as propriedades sejam atendidas, então o modelo está correto. Caso o modelo não atenda alguma propriedade, então é gerado um contra-exemplo mostrando o porquê da sua invalidade.

Formalmente, representa-se o modelo com uma estrutura *Kripke*, que consiste em um conjunto de estados, um conjunto de transições entre estados e uma função que determina, para cada estado, um conjunto de propriedades que são verdadeiras (CLARKE; GRUMBERG; PELED, 1999). Por exemplo, um modelo que possui três variáveis *booleanas* $a$, $b$ e $c$, então $(a = 1, b = 1, c = 1)$, $(a = 1, b = 0, c = 1)$ e $(a = 1, b = 0, c = 0)$ são possíveis estados do modelo. A representação simbólica desses estados seria: $(a, b, c)$, $(a, \overline{b}, c)$ e $(a, \overline{b}, \overline{c})$ respectivamente, onde $a$ significa que a variável é verdadeira e $\overline{a}$ significa que a variável é falsa.

Fórmulas *booleanas* podem ser verdadeiras ou falsas em um determinado estado. Por exemplo, a fórmula $a \vee c$ é verdadeira em todos os três estados apresentados acima. Também pode-se representar um estado através de uma fórmula *booleana*. Por exemplo, o estado $(a, \overline{b}, c)$ é representado pela fórmula $a \wedge \neg b \wedge c$.

Transições também podem ser representadas por fórmulas *booleanas*. A transição $s \rightarrow t$ é representada por dois conjuntos distintos de variáveis, um para o estado corrente $s$ e outro para o próximo estado $t$. Para cada variável do conjunto do estado corrente existe uma variável correspondente no conjunto do próximo estado. Por exemplo, se as variáveis do estado corrente são $a$, $b$ e $c$, então as variáveis do próximo estado serão $a'$, $b'$ e $c'$. Seja $f_s$ a fórmula *booleana* para o estado $s$ e $f_t$ a fórmula *booleana* para o estado $t$, então a transição $s \rightarrow t$ será representada por $f_s \wedge f_t$. A Figura 8 apresenta um exemplo

da transição do estado $(\overline{a}, \overline{b}, \overline{c})$ para o estado $(\overline{a}, b, \overline{c})$, que é representado pela fórmula $(\neg a \wedge \neg b \wedge \neg c) \wedge (\neg a' \wedge b' \wedge \neg c')$.

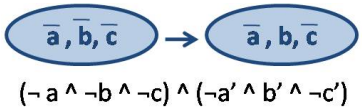



Figura 8: Exemplo de transição de estados e sua fórmula *booleana*

Uma fórmula *booleana* pode representar um conjunto de estados ou um conjunto de transições. Por utilizar símbolos para representar os estados e as transições, os algoritmos que utilizam essa técnica são chamados algoritmos simbólicos e a técnica é chamada Verificação Simbólica de Modelos.

A verificação simbólica de modelos já foi aplicada com sucesso na validação de grandes sistemas complexos, como por exemplo: um controlador de aeronaves (CAMPOS et al., 1994), um controlador robótico (CAMPOS et al., 1995) e um sistema de arquivos (YANG et al., 2006). Essa técnica se mostra bastante eficiente por ser capaz de realizar a verificação em modelos com mais de $10^{30}$ estados em apenas alguns segundos. A chave de toda essa eficiência é a utilização de diagramas binários de decisão (BDD - *Binary Decision Diagrams*) para representar o grafo de transição de estados e para verificar se a propriedade é válida ou não.

### 3.2.1 BDD - Diagrama binário de decisão

Diagrama binário de decisão (BDD) é uma representação canônica para fórmulas *booleanas*. Um BDD é obtido através da simplificação de uma árvore binária de decisão que representa a fórmula *booleana*. A simplificação é realizada através da combinação de todas as sub-árvores isomórficas e da eliminação de todos os nós com filhos isomórficos. O resultado da simplificação é um grafo acíclico direcionado. A Figura 9 ilustra a árvore binária de decisão e o BDD correspondente para a fórmula $(a \wedge b) \vee (c \wedge d)$.

Formalmente, o BDD é um grafo acíclico direcionado com dois tipos de vértices: não-terminais e terminais. Cada vértice não terminal $v$ é rotulado por $var(v)$, e representa uma das variáveis *booleanas* da fórmula. Com exceção do vértice raiz, cada vértice $v$ tem pelo menos uma aresta de entrada. Cada vértice $v$ também tem duas arestas de saída: $left(v)$, que corresponde ao caso em que $var(v) = 0$, e $right(v)$, que corresponde ao caso em que $var(v) = 1$.

Um BDD tem dois vértices terminais rotulados 0 e 1, que representam os valores $falso$

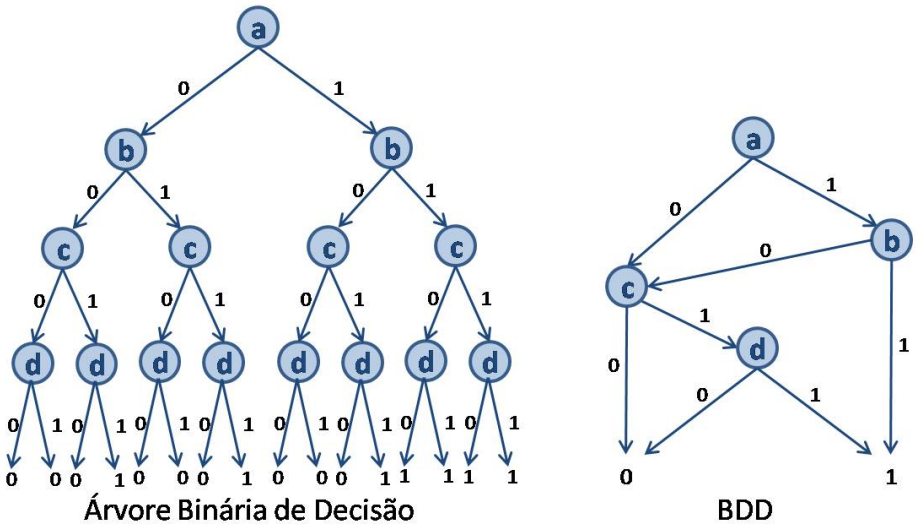



Figura 9: Árvore binária de decisão e BDD correspondente para a fórmula $(a \wedge b) \vee (c \wedge d)$

e *verdadeiro*, respectivamente. Para cada associação de valores às variáveis *booleanas* da fórmula, existe um caminho no BDD partindo do vértice raiz para um dos vértices terminais. Se o caminho terminar no vértice de rótulo 0, então a fórmula não foi satisfeita, da mesma forma, se o caminho terminar no vértice de rótulo 1 a fórmula foi satisfeita.

BDD é uma representação canônica para fórmulas *booleanas*, o que significa dizer que duas fórmulas são logicamente equivalentes se e somente se tiverem BDDs isomórficos. Isso simplifica a execução de operações frequentes sobre fórmulas *booleanas*, tais como: verificar a equivalência de fórmulas ou verificar se uma fórmula é satisfatível ou não.

Entretanto, BDD também tem suas desvantagens. A mais significativa diz respeito a ordem em que as variáveis aparecem na árvore de decisão e consequentemente no BDD. Para uma fórmula booleana, o tamanho do BDD está diretamente relacionado a ordenação das variáveis. O BDD pode crescer de linear para exponencial dependendo do número de variáveis existentes na fórmula. Além disso, o problema de ordenar as variáveis de forma a minimizar o BDD é um problema NP-completo. Apesar de existirem heurísticas para ordenar as variáveis da melhor maneira possível, em alguns casos é necessário realizar a ordenação manualmente (BRYANT, 1986).

### 3.2.2 Modelando sistemas concorrentes

Formalmente, representa-se o modelo com uma estrutura *Kripke* (CLARKE; GRUMBERG; PELED, 1999), $M = (S, S_0, R, L)$ em que:

1. $S$ é um conjunto de estados;

2. $S_0 \subseteq S$ é um conjunto de estados iniciais;

3. $R \subseteq S \times S$ é uma relação transição, que deve ser total, ou seja para cada $s \in S$ existe um $s' \in S$ em que $R(s, s')$;

4. $L : S \rightarrow 2^{AP}$ é uma função que rotula cada estado com um conjunto de proposições atômicas que são verdadeiras para o estado. Lembrando que uma proposição atômica é uma expressão na forma $v = d$, onde $v \in V$ é uma variável do conjunto de variáveis do sistema e $d \in D$ é um domínio possível para a variável.

Um caminho na estrutura *Kripke* $M$ partindo do estado $s$ é uma sequência infinita de estados $\pi = s_0 s_1 s_2 \ldots s_N$ em que $s_0 = s$ e $R(s_i, s_{i+1})$ é válido para todo $i \geq 0$.

Para ilustrar essa notação, considere um simples exemplo onde $V = \{x, y\}$, $D = \{0, 1\}$, $S_0(x, y) \equiv (x = 0) \wedge (y = 1)$ e a única transição possível é $x = y$, representada pela fórmula $R(x, y, x', y') \equiv (x' = y) \wedge (y' = y)$. A estrutura *Kripke* $M = (S, S_0, R, L)$ correspondente para essa fórmula é:

- $S = \{(0,0), (0,1), (1,0), (1,1)\}$.
- $S_0 = \{(0,1)\}$.
- $R = \{[(0,0), (0,0)], [(0,1), (1,1)], [(1,0), (0,0)], [(1,1), (1,1)]\}$.
- $L((0,0)) = \{x = 0, y = 0\}$, $L((0,1)) = \{x = 0, y = 1\}$, $L((1,0)) = \{x = 1, y = 0\}$ e $L((1,1)) = \{x = 1, y = 1\}$.

A Figura 10 apresenta graficamente a estrutura *Kripke* $M$. Note que o único caminho possível partindo do estado inicial é $(0,1)(1,1)(1,1)(1,1)\ldots(1,1)$. Ou seja, essa é a única computação permitida pelo sistema.

### *3.2.3 Lógica Temporal*

Lógica temporal é um formalismo muito utilizado para descrever uma sequência de transições entre estados (CLARKE; GRUMBERG; PELED, 1999). A lógica temporal permite raciocinar sobre o sistema em termos de ocorrências de eventos, por exemplo: pode-se pensar se um determinado evento irá *eventualmente* ocorrer ou se um outro evento irá *sempre* ocorrer.

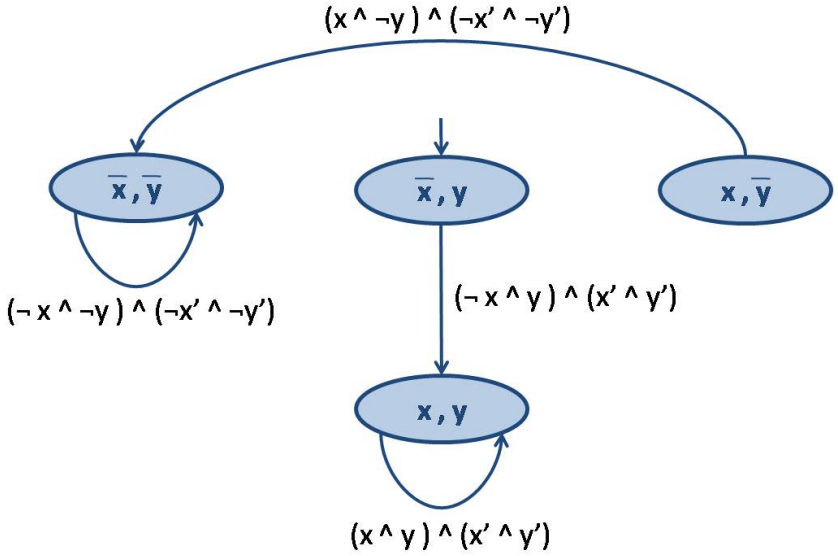



Figura 10: Grafo de transição de estados para a estrutura *Kripke* que representa a fórmula $x = y$.

Existem várias propostas para utilização de lógica temporal. Bellini, Mattolini e Nesi (2000) propõe em seu artigo uma taxonomia que divide as lógicas de acordo com sua estrutura temporal (*linear* ou *ramificada*) e suas características temporais (*contínuo* ou *discreto*). Na lógica linear, o tempo é visto como uma cadeia contínua de momentos temporais. Já na lógica ramificada, podem existir vários futuros possíveis em um momento temporal, conforme mostra a Figura 11. O tempo é contínuo sempre que entre dois momentos temporais existir outro momento temporal; e o tempo é discreto quando não é possível determinar outro momento temporal entre dois momentos temporais. Neste trabalho utiliza-se a lógica temporal ramificada e discreta chamada Lógica de Árvores de Computação ou CTL, do inglês *Computational Tree Logic*.

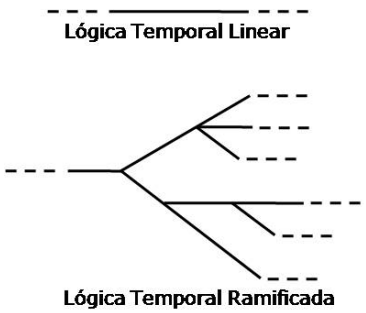



Figura 11: Estrutura temporal *linear* e *ramificada*

CTL é uma lógica utilizada para expressar propriedades que serão validadas por um verificador de modelos. As árvores de computação utilizadas nessa lógica são derivadas do grafo de transição de estados definidos pela estrutura *Kripke*. O grafo de transição de estados é transformado em uma árvore infinita cuja raiz é o estado inicial do grafo. Veja

um exemplo na Figura 12. Os caminhos na árvore de computação representam todas as computações possíveis no modelo.

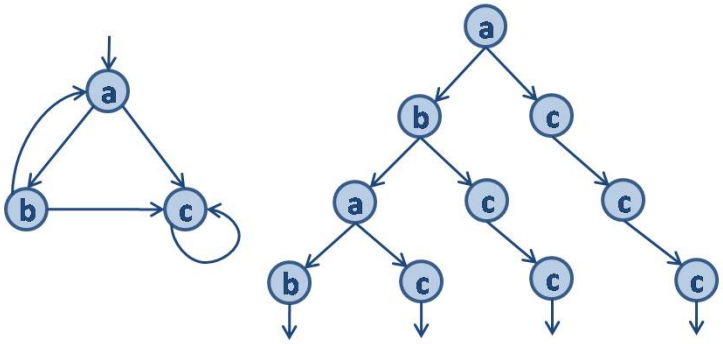



Figura 12: Grafo de transição de estados e sua respectiva árvore de computação.

A linguagem CTL possui operadores que podem ser aplicados sobre os caminhos de uma árvore de computação. Em uma fórmula CTL os operadores devem aparecer aos pares e na seguinte ordem: um *quantificador de caminho* seguido de um *operador temporal*. O quantificador de caminho define o escopo de caminho em que a fórmula $f$ deve ser verdadeira. Existem dois quantificadores de caminho: **A** (*all paths*), que significa para todo caminho; e **E** (*some path*), que significa que existe um caminho. Os operadores temporais, por sua vez, definem o comportamento temporal que deve ocorrer ao longo do caminho relacionado à fórmula $f$. Os operadores temporais são:

- **F** (*in the future* ou *eventually*) - iniciando da raiz da árvore de computação, $f$ deve ser verdadeira em algum estado do caminho;
- **G** (*globally* ou *always*) - iniciando da raiz da árvore de computação, $f$ deve ser verdadeira em todos os estados do caminho;
- **X** (*next time*) - iniciando da raiz da árvore de computação, $f$ deve ser verdadeira no segundo estado do caminho.
- **U** (*until*) - significa que existe um estado $s$ no caminho onde a fórmula $g$ é válida, e que em todos os estados anteriores a $s$, $f$ é valido;
- **R** (*release*) - iniciando da raiz da árvore de computação, $f$ deve ser verdadeira em todos os estados do caminho até que a fórmula $g$ seja verdadeira, quando então $f$ pode deixar de ser verdadeira. Entretanto, $g$ pode nunca ocorrer;

Dessa forma, uma fórmula CTL pode ser definida como:

1. Se $p \in AP$, então $p$ é uma fórmula CTL. Lembrando que $AP$ é um conjunto de proposições atômicas.

2. Se $f$ e $g$ são fórmulas CTL, então $\neg f$, $f \vee g$, $f \wedge g$, $AFf$, $EFf$, $AGf$, $EGf$, $AXf$, $EXf$, $A[fUg]$, $E[fUg]$, $A[fRg]$ e $E[fRg]$ são também fórmulas CTL.

Considerando a estrutura *Kripke* $M = (S, p, L)$ [1], a notação para $M$ que satisfaz a fórmula CTL $f$ a partir do estado estado $s \in S$ é:

$$M, s \models f$$

Seja $f$ e $g$ fórmulas CTL, a relação de satisfabilidade $\models$ é definida indutivamente como:

$M, s \models p$ $\Leftrightarrow p \in L(s)$
$M, s \models \neg f$ $\Leftrightarrow M, s \not\models f$
$M, s \models f \vee g$ $\Leftrightarrow M, s \models f$ ou $M, s \models g$
$M, s \models f \wedge g$ $\Leftrightarrow M, s \models f$ e $M, s \models g$
$M, s \models AFf$ $\Leftrightarrow$ para todos os caminhos partindo de $s, s_k \in S$ é alcançável e $s_k \models f$
$M, s \models EFf$ $\Leftrightarrow$ existe um caminho partindo de $s, s_k \in S$ é alcançável e $s_k \models f$
$M, s \models AGf$ $\Leftrightarrow$ para todos os caminhos $\pi = s_0 s_1 s_2 \ldots, s_i \models f$, para todo $i \geq 0$, e $s_0 = s$
$M, s \models EGf$ $\Leftrightarrow$ existe um caminho $\pi = s_0 s_1 s_2 \ldots, s_i \models f$, para todo $i \geq 0$, e $s_0 = s$
$M, s \models AXf$ $\Leftrightarrow$ para todo $s_x$ tal que $p(s, s_k)$ seja definido, $s_k \models f$
$M, s \models A[fUg]$ $\Leftrightarrow$ para todo caminho $\pi = s_0 s_1 s_2 \ldots s_k \ldots, s_i \models f$, para todo $0 \leq i < k$ e $s_k \models g$
$M, s \models E[fUg]$ $\Leftrightarrow$ existe um caminho $\pi = s_0 s_1 s_2 \ldots s_k \ldots, s_i \models f$, para todo $0 \leq i < k$ e $s_k \models g$

Apesar de existirem várias combinações possíveis de quantificadores de caminho e operadores temporais, é possível expressar qualquer fórmula CTL apenas com os operadores $EX$, $EU$ e $EG$ (CLARKE; GRUMBERG; PELED, 1999) seguindo as seguintes transformações:

- $AXf = \neg EX \neg f$
- $EFf = E[true\, U f]$
- $AGf = \neg EF \neg f$
- $AFf = \neg EG \neg f$

[1] Nesse momento não importa o conjunto de estados iniciais $S_0$.

- $A[fUg] \equiv \neg E[\neg gU(\neg f \wedge \neg g)] \wedge \neg EG\neg g$
- $A[fRg] \equiv \neg E[\neg fU\neg g]$
- $E[fRg] \equiv \neg A[\neg fU\neg g]$

A Figura 13 apresenta graficamente as mais frequentes operações em linguagem CTL. Alguns exemplos típicos de fórmulas CTL são:

- $EF(started \wedge \neg ready)$ - é possível encontrar um estado em que a propriedade *started* é válida e a propriedade *ready* não é válida?
- $AG(req \rightarrow AF(ack))$ - é sempre válido que se o sinal *req* estiver ativo, então, eventualmente no futuro o sinal *ack* também estará ativo?
- $A[luzVerde\,U\,moveBraco]$ - é sempre válido que o braço de robô se movimenta quando a luz verde é acionada?

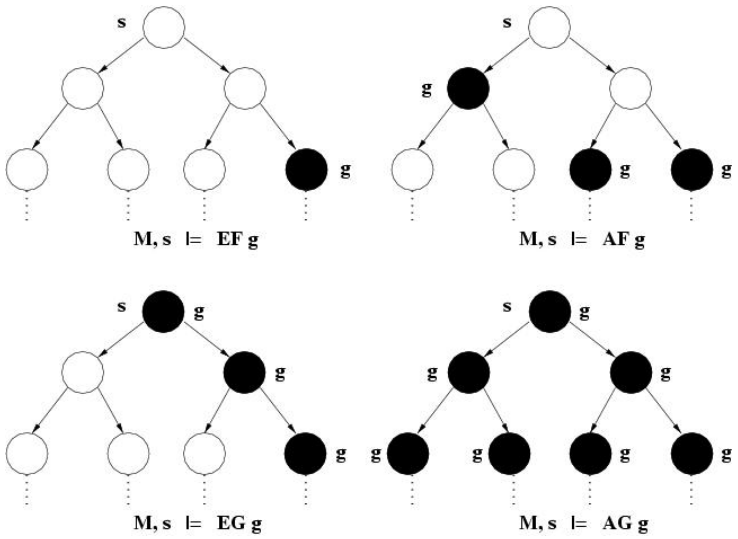



Figura 13: Operações básicas em lógica CTL em uma árvore de computação. Os estados com $s$ são os estados iniciais. Os estados na cor preta representam os estados em que a proposição $g$ é verdadeira.

## 3.3 A Linguagem SMV

Esta seção apresenta a linguagem SMV (*Symbolic Model Verifier*) (MCMILLAN, 1992) base do verificador NuSMV utilizado neste trabalho. Uma descrição mais completa do verificador pode ser encontrada em `http://nusmv.irst.itc.it/`.

### 3.3.1 Introdução

*Symbolic Model Verifier* (SMV) é uma ferramenta de verificação de especificações descritas em lógica temporal CTL sobre sistemas de estados finitos. A linguagem de entrada do SMV foi concebida para permitir a descrição de sistemas de estados finitos síncronos ou assíncronos. A linguagem permite a descrição de módulos hierárquicos e a definição de componentes reutilizáveis. O tipo de dado básico da linguagem é o tipo escalar finito, mas a linguagem também permite a criação de estruturas de dados estáticos.

A lógica CTL permite descrever as propriedades temporais de forma rica, precisa e concisa. O verificador utiliza os algoritmos de verificação baseados em OBDD (*Ordered Binary Decision Diagrams*) para determinar com eficiência quando uma especificação expressa em linguagem CTL é ou não satisfeita.

O objetivo principal da linguagem de entrada do verificador SMV é proporcionar uma descrição simbólica das relações de transição de estados de uma estrutura *Kripke*. Qualquer fórmula de proposições pode ser utilizada para descrever a relação de transição, o que provê à linguagem grande flexibilidade e ao mesmo tempo um certo risco de inconsistências. Por exemplo, a presença de contradições lógicas pode resultar em um *deadlock*, ou seja, um ou mais estados sem nenhum sucessor.

Um programa SMV pode ser visto como um conjunto de equações simultâneas, cuja solução determina o próximo estado. O compilador assegura que o programa possui apenas uma atribuição para cada variável e não possui dependências circulares ou erros de tipagem.

A seguir, são apresentados alguns exemplos que ilustram os conceitos básicos da linguagem.

### 3.3.2 Arquivo de entrada

Considere o seguinte exemplo de código:

```
MODULE main
VAR
   request : boolean;
   state : { ready, busy };
ASSIGN
   init(state) := ready;
```

```
    next(state) := case
       state = ready & request : busy;
       1 : { ready, busy };
    esac;
SPEC AG(request -> AF state = busy)
```

O exemplo descreve um modelo e uma especificação em lógica CTL. O modelo é a estrutura *Kripke*, cujos estados são definidos por uma coleção de variáveis, que podem ser *booleanas* ou do tipo escalar. As variáveis são declaradas logo após a instrução $VAR$. No exemplo são declaradas duas variáveis: *request*, que é uma variável *booleana*, e *state*, que é uma variável do tipo escalar, que pode receber os valores *ready* ou *busy*.

Os valores das variáveis escalares são codificadas pelo compilador utilizando coleções de variáveis *booleanas*, assim, a transição de estados continua sendo representada por um BDD. Essa codificação de variáveis escalares é transparente para o usuário do sistema.

As relações de transição da estrutura *Kripke* e os estados iniciais são determinadas por um conjunto de atribuições paralelas, que são codificadas após a instrução $ASSIGN$.

A instrução *init* inicializa uma variável. No exemplo, a variável *state* é inicializada com o valor *ready*. O próximo valor de uma variável é determinado pela instrução *next*. No exemplo, o próximo valor da variável *state* depende do valor corrente das variáveis de acordo com a expressão *case*:

```
case
    state = ready & request : busy;
    1 : { ready, busy };
esac;
```

O valor da expressão *case* é determinado pela primeira expressão do lado direito dos dois pontos (:) desde que a condição do lado esquerdo seja verdadeira. Portanto, se a variável $state = ready$ e a variável $request = true$, então o resultado da expressão será *busy*, caso contrário, o resultado é o conjunto de valores $\{ready, busy\}$. Quando um conjunto é atribuído para uma variável, o verificador realiza uma escolha não-determinística entre os valores do conjunto e atribui para a variável. Portanto, caso $state \neq ready$ ou $request = false$, o próximo valor da variável *state* pode tanto ser *ready* ou *busy*.

Escolhas não-determinísticas são úteis para descrever sistemas que ainda não estão completamente implementados, ou para abstrair modelos complexos, em que o valor de algumas variáveis não pode ser precisamente determinado.

Note que no programa não é atribuído nenhum valor (seja ele valor inicial ou próximo valor) para a variável *request*. Desta forma, o verificador irá escolher livremente o valor da variável *request*, que tem a característica de ser uma entrada livre do sistema.

A propriedade a ser verificada no modelo aparece como uma formula CTL definida pela instrução *SPEC*:

```
SPEC AG(request -> AF state = busy)
```

O verificador irá percorrrer todos os estados alcançáveis verificando se a propriedade é satisfeita ou não. No exemplo, a propriedade sendo validada é: para todos os estados do modelo, sempre que a variável *request* for verdadeira, haverá algum estado posterior em que a variável *state* será igual a *busy*?

### 3.3.3 Módulos reutilizáveis

O próximo exemplo de código ilustra a definição de um módulo reutilizável. O modelo representa um contador binário de 3 bits. Observe que o módulo de nome *main* tem um significado especial para o compilador, uma vez que é esse o módulo executável do arquivo de entrada. A ordem em que os módulos são definidos no programa não interfere na sua execução.

```
MODULE main
VAR
     bit0 : counterCell(1);
     bit1 : counterCell(bit0.carryOut);
     bit2 : counterCell(bit1.carryOut);
SPEC
     AG AF bit2.carryOout

MODULE counterCell(carryIn)
VAR
     value : boolean;
ASSIGN
     init(value) := 0;
     next(value) := value + carryIn mod 2;
DEFINE
     carryOut := value & carryIn;
```

Através do exemplo, percebe-se que uma variável pode ser declarada como uma instância de um módulo definido pelo usuário. No código em questão, o módulo é o *counterCell*, que é então instanciado três vezes para as variáveis *bit*0, *bit*1 e *bit*2.

O módulo *counterCell* possui um parâmetro de entrada chamado *carryIn*. Para a variável *bit*0, esse parâmetro de entrada possui o valor 1. Já na variável *bit*1, o parâmetro de entrada informado é igual a expressão definida *carryOut* do módulo atribuído ao *bit*0. O mesmo ocorre com a variável *bit*2, que recebe como parâmetro de entrada o valor da expressão *carryOut* do módulo atribuído ao *bit*1.

A instrução $DEFINE$ é utilizada para atribuir a expressão $value \,\&\, carryIn$ para o símbolo *carryOut*. O efeito desta instrução é análogo ao conjunto de instruções:

```
VAR
    carryOut : boolean;
ASSIGN
    carryOut := value & carryIn;
```

Observe, porém, que o valor corrente das variáveis é que são aplicados a expressão de *carryOut* e não o próximo valor das variáveis. Símbolos definidos com a instrução $DEFINE$ são muitas vezes mais indicados por não introduzirem mais uma variável à representação BDD do modelo. Por outro lado, esses símbolos não podem receber valores não-determinísticos.

### *3.3.4 A instrução TRANS*

A relação de transição de estados $R$ de um modelo é um conjunto de pares, sendo que cada par é formado por um estado origem e um estado destino. A instrução $TRANS$ é utilizada para declarar uma expressão booleana que representará a relação de transição de estados $R$. A sintaxe da instrução $TRANS$ é:

```
decl :: TRANS expr
```

Será considerado um erro se o resultado da expressão for qualquer valor diferente de 0 ou 1. Caso exista mais de uma instrução $TRANS$, a relação de transição será a conjunção de todas as instruções $TRANS$.

A relação de transição de estados $R$ deve ser total. Nesse sentido, a instrução $TRANS$ pode gerar inconsistências no programa que serão detectadas e reportadas pelo compilador.

### *3.3.5 Contra-exemplo*

Se alguma propriedade do modelo não for atendida, o verificador apresenta um contra-exemplo que prova que a propriedade é falsa. Isso nem sempre é possível, já que fórmulas precedidas pelo quantificador de caminho $E$ não podem ser provadas falsas apenas apresentando um único caminho em que a especificação é falsa. Por analogia, fórmulas precedidas pelo quantificador de caminho $A$ não podem ser provadas verdadeiras apenas apresentando um único caminho em que a especificação é verdadeira. Além disso, algumas fórmulas requerem infinitos caminhos de execução como contra-exemplos. Nesse caso, o verificador de modelos apresenta um caminho cíclico (*em loop*) incluindo o estado inicial do ciclo.

# 4 ABORDAGEM

O problema de identificação das vítimas de desastres em larga escala utilizando inferência genética envolve um grande processamento computacional e requer algoritmos eficientes e precisos. Em um cenário típico, existem centenas de amostras de DNA colhidas a partir dos restos mortais, que devem ser comparadas ao DNA dos familiares das vítimas. Além disso, as amostras colhidas no local do desastre podem estar degradadas devido a exposição ao sol, fogo ou água. Uma solução robusta para o problema deve considerar a qualidade do DNA coletado, comparar o DNA de cada amostra com cada família e apresentar os resultados.

Após a coleta, o perfil genético deve ser obtido de cada uma das amostras. Nesse momento, é importante determinar o número de locos de STR que serão extraídos de cada amostra e que farão parte do perfil genético das pessoas. No caso da amostra da vítima estar extremamente degradada, ocorre do perfil genético ser apenas parcial, ou seja, o perfil genético pode não ter os alelos definidos para todos os locos testados.

Para cada família deve ser realizada uma análise da estrutura familiar e um heredograma deve ser montado. Preferencialmente, deve-se coletar amostras de ambos os pais da vítima. Caso o perfil genético de ambos os pais não esteja disponível para testes, deve-se coletar amostras da(o) esposa(o) da vítima e de seus filhos. E caso a pessoa não seja casada ou não possua filhos, deve-se então coletar amostras de seus irmãos (ALONSO et al., 2005; LAU; TAN; TAN, 2005).

Considerando que existem $N$ famílias com o heredograma mapeado e alguns perfis genéticos conhecidos, e $M$ amostras de vítimas do desastre, o problema principal é confrontar a informação genética obtida nas amostras de cada vítima com aquelas obtidas nas amostras de cada família.

Normalmente esse problema é abordado em duas etapas. Na primeira etapa, as amostras são reunidas em grupos com o mesmo perfil genético, cada grupo representando uma vítima. Isso reduz o problema de confrontar $M$ amostras a $N$ famílias a um

problema menor: confrontar $J$ grupos de amostras a $N$ famílias, onde $J$ é um número muito menor que $M$. Na segunda etapa, as amostras agrupadas são então comparados às famílias (LIN; MYERS; XING, 2006).

Este trabalho se propõe a auxiliar na segunda etapa do problema. A seção seguinte apresenta a solução proposta.

## 4.1 Solução proposta

A solução proposta cria, para cada loco sendo testado, um modelo SMV para cada família. Por exemplo, se 13 locos serão testados para 1.000 famílias e vítimas, serão criados 13.000 modelos SMV, sendo um modelo para cada loco de cada família. Depois de gerados, todos os modelos são automaticamente submetidos ao verificador NuSMV. O resultado da verificação é analisado e uma classificação das vítimas com maior compatibilidade genética é montada para cada uma das famílias testadas.

Uma outra opção seria criar apenas um modelo SMV para cada família e nele realizar a verificação para todos os locos. Simulações foram realizadas com essa estratégia e se mostraram ineficientes. Com essa abordagem, o número de variáveis declaradas no modelo é elevado devido a quantidade de locos sendo simultaneamente testados. Como cada variável pode assumir vários valores, a quantidade de estados alcançáveis no modelo aumenta consideravelmente, o que inviabiliza a verificação devido ao problema de explosão de estados. Por exemplo, considerando que para cada loco testado em uma família são declaradas 6 variáveis e que para representar cada variável são gastos 4 *bits* em média. Para testar 13 locos são declaradas 78 variáveis, representadas por 312 bits, o que corresponde a $2^{312}$ estados alcançáveis no modelo. Com essa quantidade de estados o verificador não consegue resolver o modelo devido ao problema de explosão de estados.

Uma outra questão que é facilitada ao se criar um modelo por loco por família é a contagem do número de locos compatíveis da família com cada vítima. Com um modelo por loco, uma vítima pode ter um loco compatível com uma família e outro loco não compatível, ao passo que com apenas um modelo, ou a vítima é compatível ou não é compatível com a família.

Desta forma, a solução de criar um modelo por loco por família se mostrou mais adequada. Apesar de serem gerados muitos modelos SMV, cada modelo é bastante simples e a sua verificação é rápida.

## 4.2 O modelo SMV para representar uma família

O modelo SMV proposto se baseia nos princípios mendelianos da genética, que são representados pela fórmula *booleana*:

$$((F_1 = P_1 \vee F_1 = P_2) \wedge (F_2 = M_1 \vee F_2 = M_2))$$
$$\vee$$
$$((F_1 = M_1 \vee F_1 = M_2) \wedge (F_2 = P_1 \vee F_2 = P_2))$$

Onde as variáveis $F_1$ e $F_2$ representam os alelos de uma pessoa para um loco; $M_1$, $M_2$, $P_1$ e $P_2$ são, respectivamente, os alelos da mãe e do pai desta pessoa, para o mesmo loco. Caso a fórmula seja verdadeira, é possível que a pessoa tenha herdado seus alelos dos genitores testados, ou seja, a relação de parentesco segue os princípios mendelianos da genética. Caso a fórmula seja falsa, o princípio mendeliano não é válido para esse loco no trio (pai/mãe/filho) testado, o que elimina a probabilidade dessa pessoa ter herdado seus alelos desse casal, desconsiderando a ocorrência de mutações genéticas.

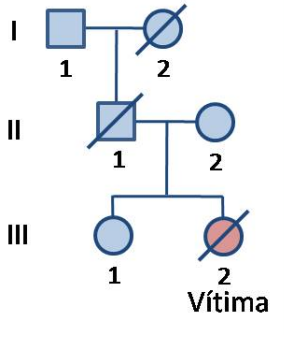
No SMV essa fórmula é representada pelo módulo *Mendel*, conforme o seguinte código:

```
MODULE Mendel (P1, P2, M1, M2, F1, F2)
TRANS
( ((F1=P1 | F1=P2) & (F2=M1 | F2=M2)) |
  ((F1=M1 | F1=M2) & (F2=P1 | F2=P2)) )
```

O módulo recebe seis variáveis que representam os alelos para um loco: $P1$ e $P2$ são os alelos do pai, $M1$ e $M2$ os alelos da mãe e $F1$ e $F2$ os alelos do filho. A instrução $TRANS$ da linguagem SMV determina todas as relações de transições possíveis entre essas seis variáveis de acordo com os princípios mendelianos da genética.

Uma relação familiar mais complexa é modelada combinando conjuntos da relação básica de um trio (pai/mãe/filho). A Figura 14 apresenta um exemplo de uma estrutura familiar mais complexa e o perfil genético das pessoas para o loco de nome CSF1PO. O perfil genético das pessoas I2 e II1 não estão disponíveis. Já a pessoa III2 é a vítima de acidente, cuja identidade deverá ser determinada.

A estrutura familiar da Figura 14 pode ser dividia em três trios (pai/mãe/filho): I1/I2/II1, II1/II2/III1 e II1/II2/III2. A Figura 15 apresenta os três trios em destaque.




| Perfil Genético | |
|---|---|
| **Pessoa** | **STR: CSF1P0** |
| I 1 | 9 , 12 |
| I 2 | Indisponível |
| II 1 | Indisponível |
| II 2 | 12 , 15 |
| III 1 | 12 , 12 |
| III 2 | Indisponível |

Figura 14: Exemplo de uma estrutura familiar completa.

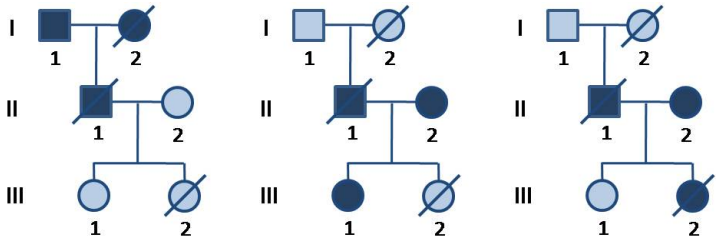



Figura 15: Exemplo de uma estrutura familiar completa destacando os trios.

Cada trio pai/mãe/filho deriva um modelo mendeliano $M_p$. O modelo final, que representa a estrutura familiar completa, é obtido da combinação dos modelos: $S = M_0 \times M_1 \times M_2 \times \ldots \times M_n$.

No SMV, o módulo *Familia* representará a estrutura familiar completa, obtida através de diversas chamadas ao módulo *Mendel*, uma chamada para cada trio pai/mãe/filho. Este módulo recebe como parâmetro duas variáveis para cada pessoa da família, representando seus alelos para o loco sendo testado. No caso da Figura 14 o código para o módulo *Familia* seria:

```
MODULE Familia (I1_1, I1_2, I2_1, I2_2,
                II1_1, II1_2, II2_1, II2_2,
                III1_1, III1_2, III2_1, III2_2)
VAR
  R1: Mendel (I1_1, I1_2, I2_1, I2_2, II1_1, II1_2);
  R2: Mendel (II1_1, II1_2, II2_1, II2_2, III1_1, III1_2);
  R3: Mendel (II1_1, II1_2, II2_1, II2_2, III2_1, III2_2);
```

No código supra $I1\_1$ e $I1\_2$ são as variáveis que representam os alelos da pessoa $I1$, $I2\_1$ e $I2\_2$ representam os alelos da pessoa $I2$ e assim sucessivamente.

O módulo *Main*, avaliado pelo verificador, declara duas variáveis para cada uma das pessoas cujo perfil genético não está disponível, inclusive a vítima. Essas variáveis representarão os alelos dessas pessoas para o loco sendo avaliado. O tipo dessas variáveis será uma enumeração com todos os valores de alelos possíveis para o loco.

Também no módulo *Main*, existirá uma chamada ao módulo *Familia*, informando como parâmetros as variáveis criadas para as pessoas sem perfil genético disponível, ou o valor do alelo para as pessoas cujo perfil genético é conhecido. O código para o módulo *Main* para o exemplo da Figura 14 é:

```
MODULE main
VAR
  I2_1 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
  I2_2 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
  II1_1 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
  II1_2 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
  III2_1 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
  III2_2 : {7, 8, 9, 10, 11, 12, 13, 14, 15};

  L1 : Familia (9,12, I2_1,I2_2, II1_1,II1_2, 12,15, 12,12, III2_1,III2_2);
```

No código são declaradas seis variáveis que representam os alelos das pessoas I2 e II1, que estão mortas, e da vítima III2. Essas variáveis são uma enumeração de 7 a 15, que são os valores de alelos possíveis para o loco CSF1PO. Se o modelo estivesse avaliando outro loco, os valores da enumeração seriam outros, dado que a faixa de alelo pode ser diferente para cada loco.

Finalmente, tem-se uma série de instruções *SPEC*, cada uma representando uma vítima do desastre sendo confrontada com a família modelada. Apenas vítimas do mesmo sexo da pessoa que é procurada pela família serão testadas, já que não faz sentido testar as vítimas do outro sexo. Isso diminui o número de verificações realizadas, melhorando o tempo de execução do sistema.

A operação *SPEC* testa se é possível encontrar no modelo um estado em que os alelos da vítima satisfazem as propriedades mendelianas para a família modelada. Se o resultado da operação *SPEC* for verdadeiro, então, para o loco sendo testado, a vítima pode fazer parte da família, caso contrário, não. Alguns exemplos de operações *SPEC* para o modelo da Figura 14 são:

```
SPEC EF ( III2_1 = 12 & III2_2 = 7 )
SPEC EF ( III2_1 = 14 & III2_2 = 14 )
```

Para a primeira instrução $SPEC$, o verificador retornará verdadeiro, já para a segunda operação, retornará falso e apresentará um contra-exemplo. O código SMV completo se encontra no Apêndice A.

Após a verificação de todos os modelos criados, uma classificação das vítimas testadas na família é construída. Essa classificação será ordenada por um percentual de compatibilidade, calculado com base na quantidade de instruções $SPEC$ que são verdadeiras para a vítima e no total de locos contidos no perfil genético da vítima. Quanto maior for o número de locos da vítima compatíveis com a família, maior será a probabilidade dela ser a vítima procurada. É importante ressaltar que o modelo não calcula probabilidades, apenas indica qual seria a vítima mais compatível com determinada família dada a quantidade de locos compatíveis.

Por exemplo, para uma vítima com 10 locos disponíveis no perfil genético a classificação do GENESys pode apontar que todos os 10 locos são compatíveis com a família ou 100% de compatibilidade. Já para uma outra vítima também com 10 locos disponíveis no perfil genético a classificação pode mostrar que apenas 5 locos são compatíveis com a família ou 50% de compatibilidade. Apenas as vítimas com 100% de compatibilidade no GENESys terão sua identidade confirmada pela rede bayesiana.

## 4.3 Base de Dados

Devido a confidencialidade do perfil genético de vítimas e famílias envolvidas em acidentes de grandes proporções, foi necessário criar uma base de dados para simular e validar o modelo proposto.

Para isso foi desenvolvido um programa capaz de gerar um conjunto de famílias e para cada família selecionar uma pessoa para ser a vítima do desastre. Cada família criada pode ter no máximo 17 pessoas conforme a estrutura apresentada pelo heredograma da Figura 16

Toda família se inicia com um casal que terá de 1 a 3 filhos. Cada um desses filhos poderá ou não se casar, mas se for casado deverá também ter de 1 a 3 filhos. A quantidade de filhos que cada casal terá é escolhida por um sorteio. A base de dados considerou que cada casal deverá ter de 1 a 3 filhos, pois de acordo com o IBGE, o número médio de

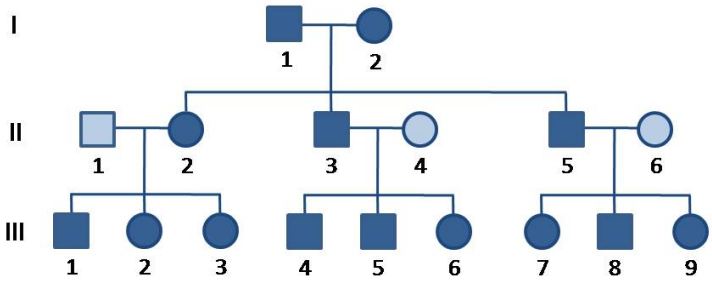



Figura 16: Estrutura máxima das famílias criadas na base de dados.

filhos por família brasileira é de 1,6 filhos (IBGE, 2001).

Ao criar a estrutura de uma família para a base de dados, todas as pessoas participantes possuem um perfil genético formado por 49 locos de STR. Para criação do perfil genético foi levado em consideração a frequência alélica para uma população.

Se os pais de uma pessoa estão representados no heredograma da família, o perfil genético dessa pessoa seguirá os princípios mendelianos da genética, ou seja, a pessoa terá um alelo herdado do pai e o outro alelo herdado da mãe, desconsiderando possíveis mutações genéticas.

As pessoas em destaque na Figura 16 são as candidatas à vítima da família. Isso se deve ao fato de que, para cada vítima, devem ser coletadas amostras dos parentes vivos mais próximos da pessoa, que no caso são: os pais, filhos e irmãos. Se por exemplo a vítima fosse a pessoa II1, a estrutura do heredograma deveria ser diferente, pois deveria representar os pais e irmãos da pessoa II1 e não os pais e irmãos da esposa de II1. A escolha da vítima da família é realizada por um sorteio entre as pessoas candidatas.

Para simular situações reais em que parentes da vítima estão mortos e não possuem o perfil genético disponível para análise, algumas pessoas do heredograma são consideradas mortas e seu perfil genético é desconsiderado na geração dos arquivos da base de dados.

A execução do programa de geração da base de dados cria um conjunto de arquivos texto que serão utilizados nas simulações. Os arquivos gerados são de três tipos:

- Arquivos de família - São arquivos que contem uma listagem das famílias que procuram vítimas do desastre.
- Arquivos de vítimas sem degradação - São arquivos que contem uma listagem das vítimas sem considerar a degradação da amostra coletada no local do desastre, ou seja, todos os locos da vítima estão disponíveis para teste.

- Arquivos de vítimas com degradação - São arquivos que contem uma listagem das vítimas considerando a degradação da amostra coletada no local do desastre. A simulação da degradação da amostra é realizada descartando alguns locos do perfil genético da vítima no momento da montagem do arquivo.

Ao todo são criados quatro arquivos de cada um dos tipos. Cada arquivo difere entre si na quantidade de locos que estão disponíveis para teste. Os arquivos possuem 13, 26, 39 e 49 locos, sendo que os arquivos com 13 locos utilizam os mesmos do sistema CODIS do FBI.

### *4.3.1 Formato do arquivo de vítimas*

A Figura 17 apresenta o formato do arquivo de vítimas.

```
 1  1  V28348  F
 2  2  V28348  D2S1780 13.0    12.0
 3  2  V28348  D3S2398 12.0    14.0
 4  2  V28348  D4S1644 14.0    13.0
 5  2  V28348  D5S2501 21.0    23.0
 6  2  V28348  D13S308 15.0    15.0
 7  2  V28348  D15S657 10.0    14.0
 8  2  V28348  D18S535 15.0    12.0
 9  1  V30980  M
10  2  V30980  CSF1PO  12.0    11.0
11  2  V30980  D2S1353 14.0    11.0
12  2  V30980  D3S2387 26.0    26.0
13  2  V30980  D3S2398 12.0    13.0
14  2  V30980  D4S2431 11.0    12.0
15  2  V30980  D5818   12.0    13.0
16  2  V30980  D7s820  10.0    11.0
17  2  V30980  D8S1130 11.0    12.0
18  2  V30980  D8S1179 16.0    15.0
19  2  V30980  D16S539 12.0    12.0
20  2  V30980  D19S433 14.0    15.0
21  2  V30980  D21S11  29.0    29.0
22  2  V30980  PENTAE  15.0    11.0
```

Figura 17: Formato do arquivo de vítimas.

No arquivo, cada vítima é representada por um conjunto de linhas. O trecho da Figura 17 apresenta duas vítimas: uma, de identificador $V28348$, é representada pelas linhas de 1 a 8; e a outra, de identificador $V30980$, pelas linhas de 9 a 22. O identificador da vítima é um número aleatório precedido pela letra $V$, que identificará o conjunto de amostras coletadas no local do desastre para aquela vítima. Todas as linhas apresentam em sua segunda coluna o identificador da vítima.

As linhas do arquivo podem ser dividas em dois tipos, que são diferenciados pela primeira coluna da linha. Cada vítima possui apenas uma linha do Tipo 1, que define,

na terceira coluna, o sexo da vítima - $F$ para feminino e $M$ para masculino. Em seguida, cada vítima possui um conjunto de linhas do Tipo 2, que representam o seu perfil genético. Cada linha do Tipo 2 possui, na terceira coluna, o nome de um STR e, nas duas colunas seguintes, os valores dos alelos da vítima.

O trecho da Figura 17 representa um arquivo que leva em consideração a degradação da amostra, pois o número de linhas do Tipo 2 é variável para cada vítima. Por exemplo, o conjunto de amostras da vítima $V28348$ possui um nível maior de degradação que o conjunto de amostras da vítima $V30980$. Quanto maior a degradação da amostra, menor será o número de locos disponíveis no perfil genético.

Para os arquivos de vítimas que não consideram degradação da amostra, todas as vítimas terão a mesma quantidade de locos no perfil genético, ou seja, a mesma quantidade de linhas do Tipo 2 no arquivo.

### *4.3.2 Formato do arquivo de famílias*

A Figura 18 apresenta o formato do arquivo de famílias.

```
1  1  F1  P1  M   0   0   0  1
2  1  F1  P2  F   0   0   0  0
3  1  F1  P3  F   P1  P2  1  0
4  1  F1  P4  M   0   0   0  1
5  1  F1  P5  M   P3  P4  0  1
6  1  F1  P6  F   P3  P4  0  1
7  2  F1  P1  CSF1PO  12.0    7.0
8  2  F1  P4  CSF1PO  10.0    11.0
9  2  F1  P5  CSF1PO  10.0    10.0
10 2  F1  P6  CSF1PO  7.0     11.0
11 2  F1  P1  D1S1612 15.0    16.0
12 2  F1  P4  D1S1612 10.0    10.0
13 2  F1  P5  D1S1612 14.0    10.0
14 2  F1  P6  D1S1612 16.0    10.0
15 2  F1  P1  D1S1656 19.0    11.0
16 2  F1  P4  D1S1656 11.0    16.0
17 2  F1  P5  D1S1656 19.0    16.0
18 2  F1  P6  D1S1656 19.0    16.0
```

Figura 18: Formato do arquivo de famílias.

No arquivo, cada família é representada por um conjunto de linhas. O trecho do arquivo da Figura 18 apresenta a estrutura de uma única família identificada como $F1$. O identificador da família é um número sequencial precedido pela letra $F$, que identificará a família. Todas as linhas apresentam em sua segunda coluna o identificador da família.

As linhas do arquivo de família podem ser dividas em dois tipos, que são diferenciados pela primeira coluna da linha. Cada família possui um conjunto de linhas do Tipo 1 e

um conjunto de linhas do Tipo 2.

As linhas do Tipo 1 definem os membros da família e sua relação de parentesco, suas colunas representam:

- Coluna 1 - Identificador do tipo da linha;
- Coluna 2 - Identificador da família;
- Coluna 3 - Identificador da pessoa - um número sequencial precedido pela letra $P$;
- Coluna 4 - Representa o sexo da pessoa - $F$ para feminino e $M$ para masculino;
- Coluna 5 e 6 - Representam o pai e a mãe da pessoa. Se os pais não estiverem representados no heredograma da família, essas colunas estarão preenchidas com o valor 0 (zero); e caso os pais estejam representados no heredograma, as colunas estarão preenchidas com o identificador de pessoa dos pais;
- Coluna 7 - É um valor *booleano* que identifica a vítima que a família procura. Cada família terá apenas uma pessoa com essa coluna igual a 1, que é a vítima do desastre;
- Coluna 8 - É um valor *booleano* que identifica se o perfil genético da pessoa está ou não disponível. O valor 0 informa que o perfil genético não está disponível. Já o valor 1 informa que o perfil genético está disponível.

Dessa forma, as linhas apresentadas na Figura 18 representam uma família cujo heredograma é exibido na Figura 19

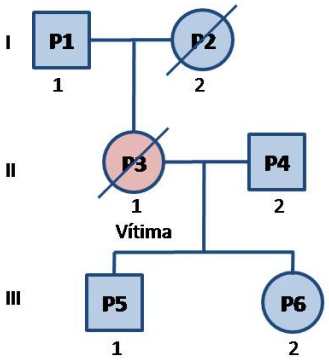



Figura 19: Heredograma descrito no trecho do arquivo de família da Figura 18.

As linhas do Tipo 2 definem o perfil genético dos membros da família. Suas colunas representam:

- Coluna 1 - Identificador do tipo da linha;
- Coluna 2 - Identificador da família;
- Coluna 3 - Identificador da pessoa - um número sequencial precedido pela letra $P$;
- Coluna 4 - Nome do loco;
- Colunas 5 e 6 - Valores dos alelos da pessoa.

## 4.4 O aplicativo GENESys

O GENESys é um aplicativo criado utilizando tecnologia JAVA. O aplicativo recebe como entrada um arquivo de vítimas e um arquivo de famílias no formato descrito na seção anterior, e, com base nesses arquivos, cria os modelos SMV para cada loco de cada família. Os modelos criados são submetidos ao verificador NuSMV e o resultado é apresentado em um arquivo de resultados.

Na Figura 20 é exibido o diagrama de classes da aplicação. A classe principal do aplicativo é chamada *Tradutor*. Ela tem a responsabilidade de receber os arquivos de entrada e iniciar o procedimento de tradução. Outras classes envolvidas são:

- *Familia*: Classe que representa o heredograma de uma família. Possui como atributos: um identificador, uma lista de pessoas que fazem parte da família e uma lista de vítimas que serão confrontadas com essa família;
- *Pessoa*: Classe que representa uma pessoa. Possui como atributos: um identificador, o sexo, uma pessoa pai, uma pessoa mãe, uma lista de locos (perfil genético) e dois *flags* que indicam se a pessoa é a vítima que a família procura e se ela está viva;
- *Vitima*: Classe que representa uma vítima do desastre. É a classe que contabiliza a quantidade de locos da vítima que são compatíveis com a família;
- *Loco*: Classe que representa um loco do perfil genético de uma pessoa. Possui como atributos: o nome do loco e os valores para o alelo1 e alelo2.

O diagrama de sequência da Figura 21 detalha o procedimento de tradução. O usuário inicia a aplicação GENESys através de seu método *main*. Como parâmetros devem ser informados:

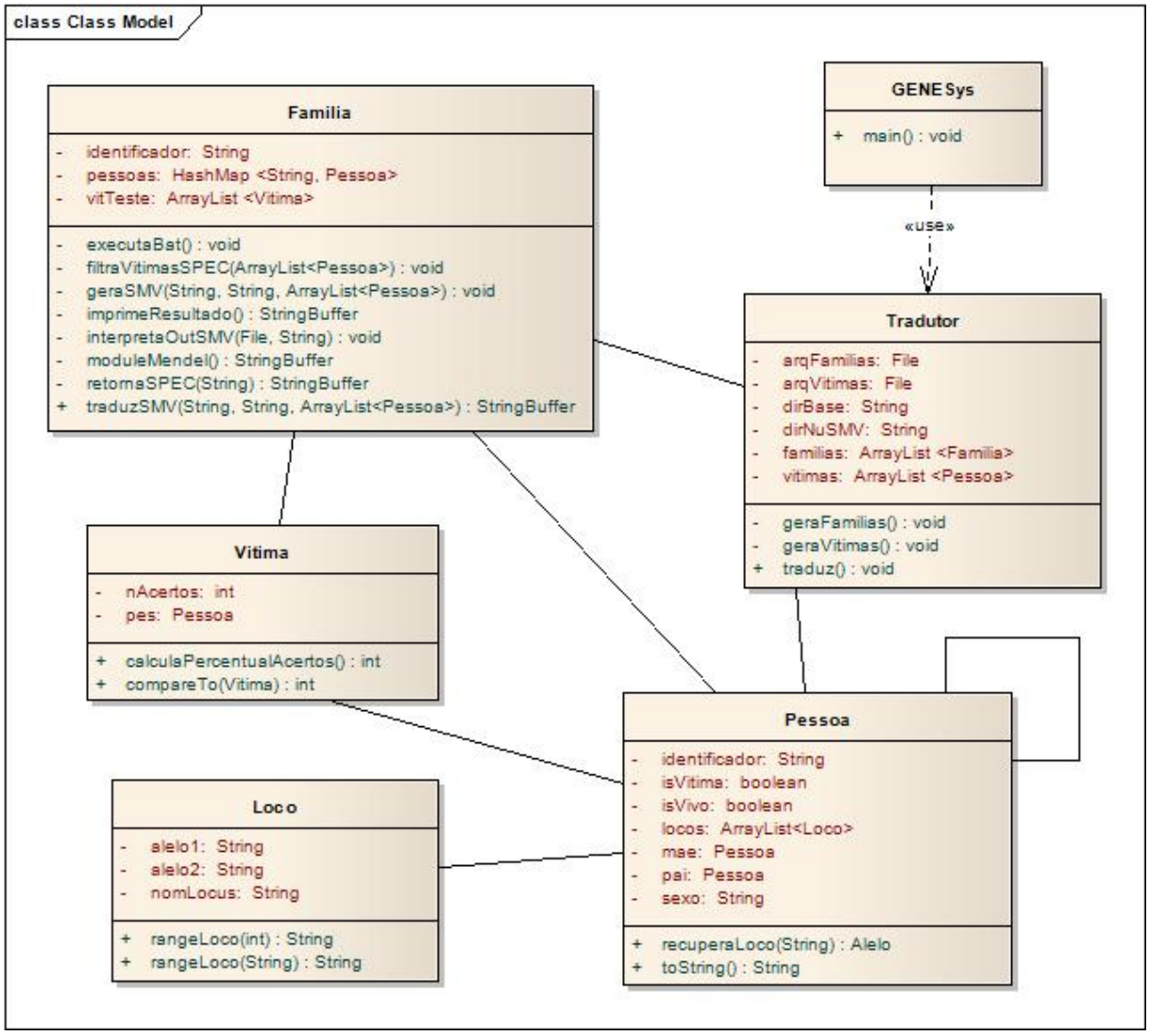



Figura 20: Diagrama de classes do GENESys.

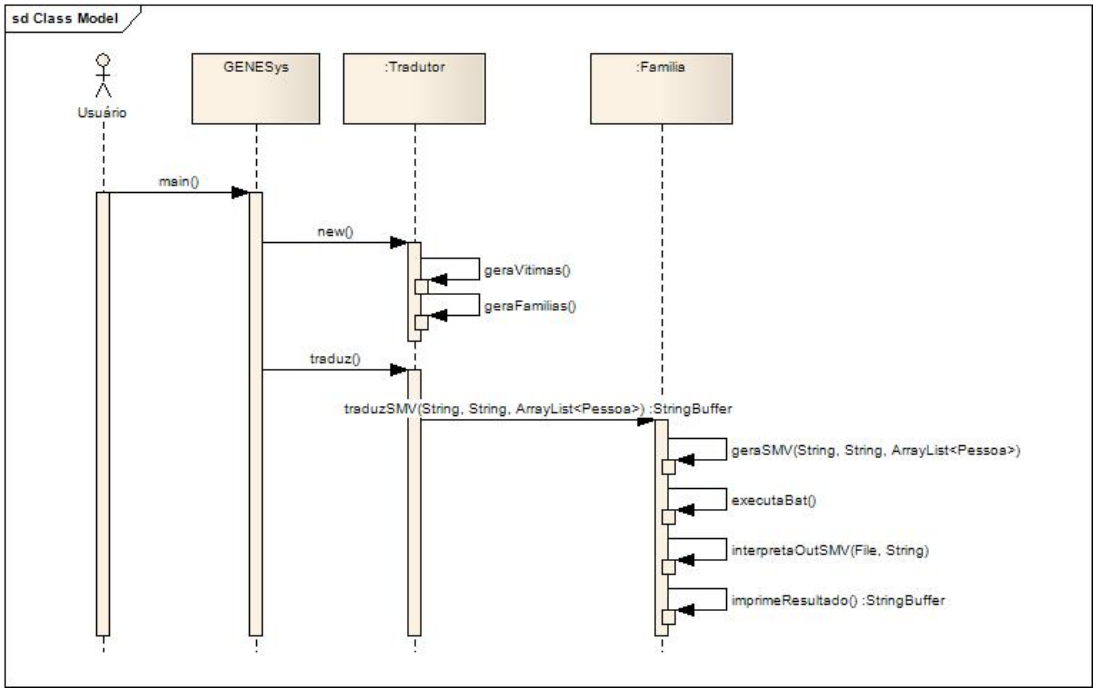



Figura 21: Diagrama de sequência da tradução realizada pelo GENESys.

- Um diretório base onde se encontram os arquivos de entrada e onde serão criados os modelos de verificação;
- O diretório de instalação do verificador NuSMV;
- O nome do arquivo de famílias;
- O nome do arquivo de vítimas.

Um exemplo de execução seria:

```
java GENESys "C:\BaseDados" "C:\NuSMV\2.4.3\bin" familia.txt vitima.txt
```

O método *main* do GENESys cria uma instância da classe *Tradutor*. Ao instanciar o Tradutor dois métodos são executados:

- *geraVitimas*: Esse método irá realizar a leitura do arquivo de vítimas, criar uma instância da classe *Pessoa* para cada vítima e armazenar essas pessoas em uma lista;
- *geraFamilias*: Esse método irá realizar a leitura do arquivo de famílias, criar uma instância da classe *Familia* para cada família e armazenar essas famílias em uma lista;

Em seguida, é chamado o método *traduz* da classe *Tradutor*. Esse método percorre a lista de famílias e, para cada uma delas, executa o método *traduzSMV* da classe *Familia*, que, por sua vez, irá realizar a tradução da família para o modelo de verificação. A tradução é realizada através dos seguintes métodos:

- *geraSMV*: Nesse método, os modelos de verificação da família (um modelo para cada loco) são criados no diretório base informado como parâmetro de execução. Um arquivo BAT é também criado, contendo uma chamada ao verificador NuSMV para cada modelo criado;
- *executaBat*: Esse método irá executar o arquivo BAT criado pelo método anterior. O resultado da execução é armazenado em um arquivo de saída;
- *interpretaOutSMV*: Esse método irá interpretar o arquivo de saída do BAT, contabilizando quais operações *SPEC* foram consideradas verdadeiras para cada vítima confrontada com a família;
- *imprimeResultado*: Finalmente, esse método irá imprimir a classificação das vítimas por ordem de compatibilidade com a família no arquivo de resultados.

```
1  FAMILIA: F1
2  V7155882 - 7 em 7 (100%)
3  V3909515 - 3 em 4 (75%)
4  V8088843 - 6 em 8 (75%)
5  V9025642 - 5 em 8 (62%)
6  V3080934 - 4 em 7 (57%)
7  V2084468 - 3 em 6 (50%)
8  V8893878 - 3 em 6 (50%)
9  V9820643 - 4 em 8 (50%)
10 V873170 - 5 em 11 (45%)
11 V9835462 - 2 em 6 (33%)
12 V8167029 - 2 em 6 (33%)
13 V4791840 - 2 em 6 (33%)
14 V8953210 - 2 em 7 (28%)
15 V186308 - 2 em 8 (25%)
16 V5133137 - 0 em 3 (0%)
17 V433073 - 0 em 3 (0%)
18
19
20 FAMILIA: F2
21 V5606784 - 6 em 6 (100%)
22 V4190392 - 3 em 5 (60%)
23 V9277544 - 4 em 7 (57%)
24 V3450592 - 4 em 9 (44%)
25 V8344110 - 3 em 8 (37%)
26 V4962692 - 3 em 8 (37%)
27 V3098813 - 2 em 8 (25%)
```

Figura 22: Formato do arquivo de resultados, considerando testes em 13 locos em amostras com degradação.

Para cada família é apresentado uma classificação de vítimas com o número de locos testado e a quantidade de locos compatíveis com a família. Um percentual de compati-

bilidade é também apresentado. O formato do arquivo de resultados é apresentado na Figura 22.

No trecho do arquivo da Figura 22 é apresentada a classificação de vítimas para as famílias $F1$ e $F2$. Para a família $F1$ a vítima $V7155882$ é a que possui maior compatibilidade genética. Para essa vítima foram testados 7 locos e todos eles obtiveram sucesso na verificação, o que significa um percentual de compatibilidade de 100%.

Todas as vítimas com percentual de compatibilidade de 100% devem ter a sua identidade confirmada através da criação de uma rede bayesiana, que calculará a probabilidade de que essa vítima seja realmente a pessoa procurada pela família.

# 5 SIMULAÇÕES

Simulações foram realizadas com bases de dados de 100, 200, 500 e 1000 famílias, considerando testes com 13, 26, 39 e 49 STR locos. Os testes com 13 locos utilizaram os mesmos locos do sistema CODIS do FBI. O aumento do número de locos testados foi realizado de 13 em 13 até o máximo de 49. Nas simulações foram consideradas amostras com e sem degradação.

Foram realizadas simulações onde:

- O número de vítimas é igual ao número de famílias - Nesse caso, existirá pelo menos uma vítima com 100% de compatibilidade genética com cada família;
- O número de vítimas é menor que o número de famílias - Nesse caso, para algumas famílias não existirá nenhuma vítima com 100% de compatibilidade;
- O número de vítimas é maior que o número de famílias - Nesse caso, existirá pelo menos uma vítima com 100% de compatibilidade genética com cada família, mas algumas vítimas não serão identificadas.

Situações em que o número de vítimas é menor ou maior que o número de famílias ocorrem quando não é possível determinar com precisão o número de vítimas do acidente. Por exemplo, em um acidente aéreo o número de vítimas é preciso e determinado pela lista de passageiros que embarcaram no avião. Já no ataque terrorista às torres do *World Trade Center* não é possível determinar com precisão quantas ou quem eram as pessoas que estavam nos prédios no momento do ataque, nesse caso o número de vítimas e de famílias podem ser diferentes.

A Figura 23 apresenta o gráfico de resultados das simulações realizadas com o número de vítimas igual ao número de famílias. Através dos gráficos observa-se uma tendência linear com um coeficiente de correlação linear de Pearson igual $0,94$, em média. O coeficiente de Pearson mede o grau de relacionamento linear entre os valores emparelhados

$x$ e $y$ em uma amostra. Quanto mais próximo de 1 for o coeficiente maior é a correlação linear (NETO, 2002).

A tendência linear é explicada pelo fato de que o aumento no número de locos testados aumentará o número de modelos gerados, e consequentemente o tempo de computação.

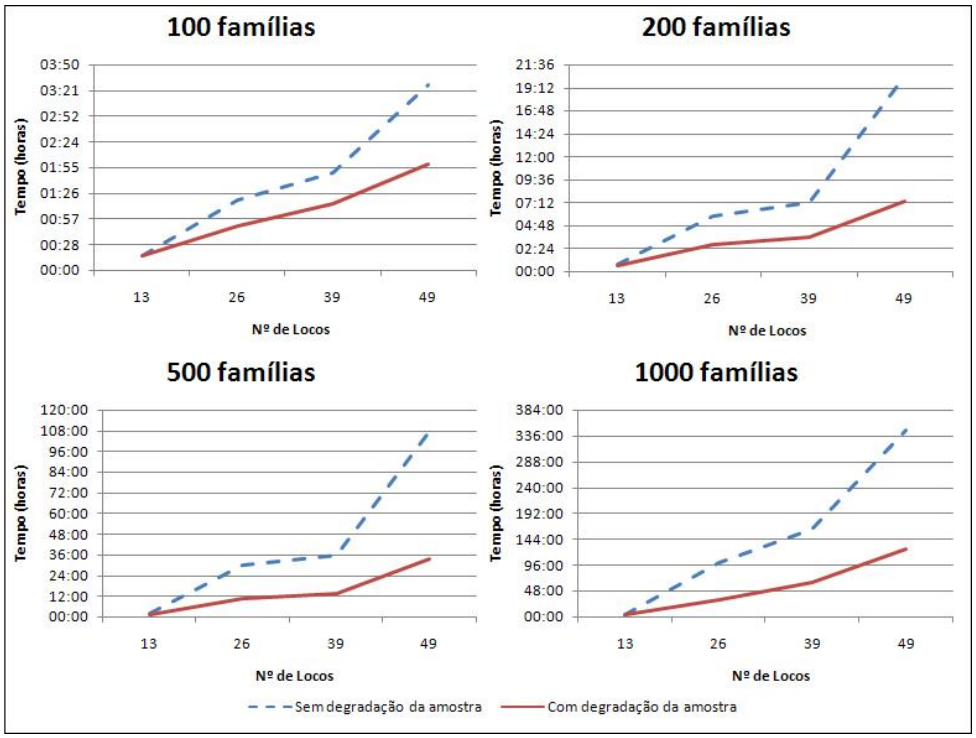



Figura 23: Resultado de simulações com número de vítimas e famílias iguais

Também observa-se que o tempo de computação considerando amostras com degradação é menor do que o tempo de computação sem considerar a degradação da amostra. Isso se deve ao fato de que com degradação a quantidade de verificações realizadas nos modelos diminui, e com isso o tempo de computação.

Já o gráfico da Figura 24 apresenta uma comparação dos resultados de simulações com número de vítimas igual, maior e menor que o número de famílias. Observa-se que o tempo de computação com 50% de vítimas a mais que o número de famílias é ligeiramente maior que o tempo de computação com o número de vítimas igual ao número de famílias. Isso é explicado pelo aumento da quantidade de verificações que são realizadas nos modelos. Quanto maior o número de vítimas, maior será o número de verificações e consequentemente o tempo de computação. A mesma análise é válida quando o número de vítimas é menor que o número de famílias.

Para a maioria das famílias avaliadas nas simulações, foi possível identificar no conjunto de vítimas apenas uma delas com 100% de compatibilidade genética com a família.

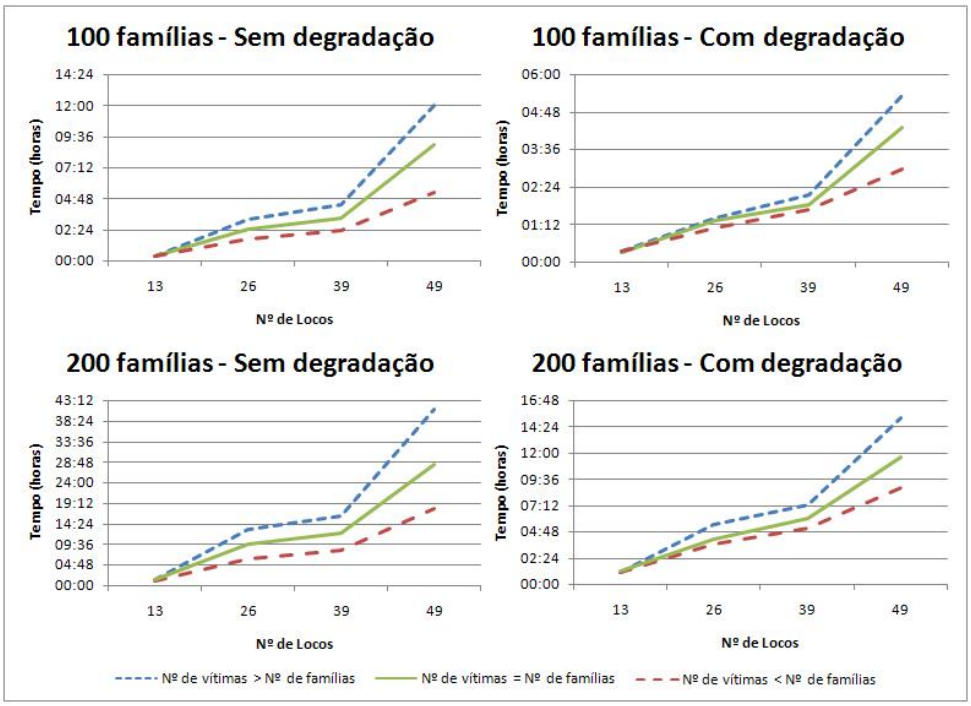



Figura 24: Comparativo de simulações com número de vítimas igual, maior e menor que o número de famílias.

Observe o exemplo do heredograma da Figura 25, onde a vítima é a pessoa I1, um viúvo, pai de três filhas. A Tabela 3 apresenta o genótipo das pessoas da família da Figura 25 para 3 locos: TPOX, VWA e FGA.

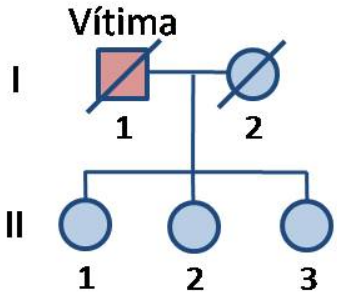



Figura 25: Heredograma de uma família com apenas uma vítima com 100% de compatibilidade genética.

Examinando-se a tabela 3, está claro que as opções de genótipo para a vítima I1 não são muitas. Por exemplo, para o loco TP0X são apenas duas as opções de genótipo para I1: (8 , 12) ou (5 , 8) . Para esse loco a vítima deve possuir pelo menos um de seus alelos igual a 8, já que uma de suas filhas (II1) é homozigótica (8 , 8). O outro alelo da vítima deve obrigatoriamente ser 5 ou 12, pois sua filha II3 não possui o alelo 8 e deve ter herdado de seu pai ou o alelo 5 ou o alelo 12. Nesse caso, se I1 for (5 , 8), sua falecida esposa obrigatoriamente deveria ser (8 , 12) e vice versa. Raciocínio parecido se aplica ao loco VWA, em que as duas opções de genótipo para I1 são (16 , 17) ou (18 , 19).

| Pessoa | TP0X | VWA | FGA |
|---|---|---|---|
| I1 | 8 , 12 / 5 , 8 | 16 , 17 / 18 , 19 | 21 , 23 / 24 , ? |
| I2 | 5 , 8 / 8 , 12 | 18 , 19 / 16 , 17 | 24 , ? / 21 , 23 |
| II1 | 8 , 8 | 16 , 18 | 21 , 24 |
| II2 | 8 , 12 | 16 , 19 | 23 , 24 |
| II3 | 5 , 12 | 17 , 18 | 21 , 24 |

Tabela 3: Genótipos das pessoas da Figura 25

Já para o loco FGA, as opções de genótipo da vítima são um pouco maiores. As opções são:

- Opção 1: Todas as filhas herdaram do pai o alelo 24. Para satisfazer as relações mendelianas, a esposa de I1 obrigatoriamente deveria ser (21 , 23). Nesse caso a vítima deveria ter pelo menos um de seus alelos igual a 24.
- Opção 2: Todas as filhas herdaram o alelo 24 da mãe. Para satisfazer as relações mendelianas, a vítima I1 deveria obrigatoriamente ser (21 , 23).
- Opção 3: As filhas herdaram o alelo 24 hora da mãe e hora do pai. Para satisfazer as relações mendelianas, um dos pais deveria ser (21 , 24) e o outro (23 , 24). Esses genótipos são um subconjunto da primeira opção, em que a vítima deve ter pelo menos um de seus alelos igual a 24.

Para esse exemplo, as opções restritas de genótipo para a vítima ocorrem para todos os locos avaliados. Por esse motivo, foi possível encontrar no conjunto de vítimas apenas uma delas com 100% de compatibilidade genética com essa família. Mas isso nem sempre é verdadeiro. Considere por exemplo o heredograma da família apresentado na Figura 26, onde a vítima é a pessoa I2, que teve apenas 1 filho (II1), que já está morto. Foram colhidas amostras do marido de I2 (I1), de duas netas de I2 (III1 e III2) e da nora de I2 (II2). A Tabela 4 apresenta o genótipo das pessoas para 3 locos: CSF1PO, FGA e D8S1179.

Aplicando-se os princípios mendelianos da genética nas informações da Tabela 4, observa-se que não é possível determinar com muita precisão qual é o genótipo da vítima. Tomando como exemplo o loco CSF1PO, sabe-se que as pessoas III1 e III2 herdaram de sua mãe (II2) o alelo 11, consequentemente o pai de III1 e III2 possui o genótipo (10 , 12). Dessa forma, II1 só pode ter herdado o alelo 10 de seu pai (I1), e, por consequência, o alelo 12 de sua mãe (I2). Assim, qualquer uma das vítimas do desastre que tenha pelo

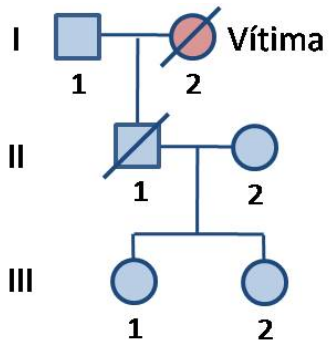



Figura 26: Heredograma de uma família com mais de uma vítima com 100% de compatibilidade genética.

| Pessoa | CSF1PO | FGA | D8S1179 |
|---|---|---|---|
| I1 | 10 , 11 | 25 , 27 | 13 , 14 |
| I2 | 12 , ? | ? , ? | 13 , ? / 14 , ? |
| II1 | 10 , 12 | 27 , ? | 13 , 14 |
| II2 | 8 , 11 | 21 , 22 | 12 , 15 |
| III1 | 10 , 11 | 22 , 27 | 12 , 13 |
| III2 | 12 , 11 | 21 , 27 | 12 , 14 |

Tabela 4: Genótipos das pessoas da Figura 26

menos um de seus alelos igual a 12 para o loco CSF1PO se encaixará no perfil genético desta família para esse loco.

Com relação ao loco FGA, sabe-se que as pessoas III1 e III2 herdaram de seu pai (II1) o alelo 27. Assim, II1 possui pelo menos um de seus alelos igual a 27 e esse alelo pode ter sido herdado de seu pai (I1). Dessa forma, não é possível chegar em nenhuma conclusão quanto ao genótipo da vítima. Para esse loco, qualquer pessoa possui compatibilidade genética com a família.

O mesmo raciocínio, quando estendido para todos os locos testados, não traz muitos esclarecimentos sobre o genótipo da pessoa I2, o que resultará em mais de uma vítima com 100% compatibilidade genética para a família. Nesse caso, a real vítima só será descoberta com a criação da rede bayesiana para todas as vítimas com 100% de compatibilidade apontadas no arquivo de resultados do GENESys.

Uma outra situação, ainda pior que a apresentada, ocorrer em desastres naturais em que várias pessoas da mesma família morrem, como foi o caso do Tsunami de 2004 ocorrido no oceano Índico. Se por exemplo todas as pessoas da família morrerem no desastre com exceção de uma, essa pessoa poderia auxiliar na identificação de seus familiares? Segundo Lau, Tan e Tan (2005), a resposta para a pergunta é algumas vezes. Por exemplo, não é muito provável que uma pessoa e seu irmão morto tenham similaridades genéticas

suficientes para que essa pessoa possa identificar o irmão. Essa situação foi comprovada em algumas simulações realizadas, em que todas as vítimas testadas pelo GENESys apontaram 100% de compatibilidade genética com a família.

Observe o caso apresentado na Figura 27, em que a vítima é a pessoa III2, uma mulher que tem apenas um irmão, uma tia e um tio vivos. A Tabela 5 apresenta o genótipo das pessoas para 3 locos: CSF1PO, D3S1358 e D1S1656. A análise da tabela através dos princípios mendelianos da genética mostra que é não é possível determinar qual é o genótipo da vítima.

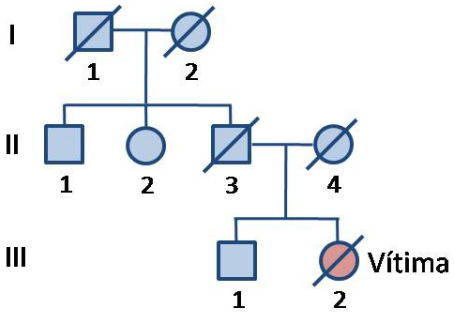



Figura 27: Heredograma de uma família em que todas as vítimas avaliadas possuem com 100% de compatibilidade genética.

| Pessoa | CSF1PO | D3S1358 | D1S1656 |
|---|---|---|---|
| I1 | 11 , ? / 11 , 13 | 17 , ? / 15 , 16 | 11 , ? / 15 , 17 |
| I2 | 11 , 13 / 11 , ? | 15 , 16 / 17 , ? | 15 , 17 / 11 , ? |
| II1 | 11 , 11 | 16 , 17 | 11 , 17 |
| II2 | 11 , 13 | 15 , 17 | 11 , 15 |
| II3 | 11 , ? / 13 , ? | 14 , ? / 17 , ? | 11 , ? |
| II4 | 13 , ? / 11 , ? | 17 , ? / 14 , ? | 11 , ? |
| III1 | 11 , 13 | 14 , 17 | 11 , 11 |
| III2 | ? , ? | ? , ? | ? , ? |

Tabela 5: Genótipos das pessoas da Figura 27

Por exemplo, para o loco CSF1PO, sabe-se que o irmão da vítima possui o genótipo (11 , 13), e que por isso um de seus pais será (11 , ?) e o outro (13 , ?). Com apenas essa informação, não é possível determinar qual é o genótipo da vítima, e a análise do genótipo dos tios vivos também não acrescenta nenhuma nova informação que auxilie na identificação da vítima. A situação se repete para os demais locos testados e nesse caso o GENESys não conseguirá eliminar nenhuma das vítimas testadas, pois todas apresentarão 100% de compatibilidade genética com a família.

Dessa forma, é possível calcular qual será o ganho em termos de número de redes bayesianas necessárias à solução do problema, com e sem a utilização do sistema

| Nº de Famílias/Vítimas | Nº de Redes Bayesianas |
|---:|---:|
| 100 | 5050 |
| 200 | 20100 |
| 500 | 125250 |
| 1000 | 500500 |

Tabela 6: Número de redes bayesianas necessárias sem utilizar o GENESys.

GENESys. Sem o GENESys, no pior caso, seriam necessárias $\sum_{i=1}^{n} i = n(n+1)/2$ redes bayesianas para a solução do problema. É o que mostra a Tabela 6.

| Nº de Famílias/Vítimas | 13 Locos | 26 Locos | 39 Locos | 49 Locos |
|---:|---:|---:|---:|---:|
| 100 | 314 | 284 | 284 | 284 |
| 200 | 1488 | 1308 | 1306 | 1304 |
| 500 | 6516 | 5431 | 5266 | 5230 |
| 1000 | 19463 | 13553 | 13031 | 12822 |

Tabela 7: Número de redes bayesianas necessárias utilizando o GENESys, sem considerar a degradação da amostra.

| Nº de Famílias/Vítimas | 13 Locos | 26 Locos | 39 Locos | 49 Locos |
|---:|---:|---:|---:|---:|
| 100 | 1191 | 521 | 376 | 338 |
| 200 | 4876 | 2225 | 1736 | 1506 |
| 500 | 27728 | 11230 | 7553 | 6471 |
| 1000 | 105663 | 37344 | 23623 | 18407 |

Tabela 8: Número de redes bayesianas necessárias utilizando o GENESys, considerando degradação da amostra.

Já as Tabelas 7 e 8 apresentam o número de redes bayesianas que seriam necessárias para solução do problema com a utilização do sistema GENESys. O aumento do número de locos testados reduz significamente o número de redes bayesianas. Por exemplo, considerando amostras com degradação, com 1.000 vítimas e famílias o número de redes necessárias para testes com 13 locos era 105.663. Já com testes em 49 locos o número de cai para 18.407 redes.

Observa-se também que o número de redes bayesianas necessárias diminui consideravelmente ao se utilizar o GENESys. Por exemplo, para 1.000 vítimas, o número de redes reduziu de 500.500 sem o GENESYS para 105.663 com a ferramenta. Se cada rede com 13 locos executa em aproximadamente 1 minuto e 10 segundos (GOMES, 2008), seriam gastos aproximadamente 14 meses para obter a resposta sem utilizar o GENESys. Com o uso da ferramenta proposta, esse tempo cai para aproximadamente 3 meses. Esse cálculo está detalhado na Tabela 9.

| | Sem o GENESys | Com o GENESys |
|---|---|---|
| Tempo de Execução do GENESys | 0 | 3 horas |
| Número de Redes Bayesianas necessárias | 500.500 | 105.663 |
| Tempo de Execução das Redes Bayesianas | 14 meses | 85,6 dias |
| Tempo Total de Execução (aproximado) | 14 meses | 86 dias |

Tabela 9: Resultado comparativo da identificação de vítimas de um acidente de 1.000 vítimas/famílias utilizando testes em 13 locos de STR com degradação da amostra e considerando que cada rede bayesiana demora 1 minuto 10 segundos para executar.

# 6 CONCLUSÕES E TRABALHOS FUTUROS

## 6.1 Conclusões

O reconhecimento de vítimas em desastres de grande escala é um processo demorado e extremamente penoso, principalmente para os familiares das vítimas, que além de perderem seus entes queridos ainda devem aguardar dias, ou até meses, para que o reconhecimento seja feito. Neste aspecto, são vários os benefícios alcançados por este trabalho do ponto de vista social, jurídico e de segurança pública.

As técnicas de identificação de vítimas podem ser classificadas pelo seu grau de complexidade. O primeiro, de menor complexidade, é a identificação por impressão digital, pelas vestimentas e feições. O segundo em grau de complexidade é a identificação por comparação com dados antropológicos e odontológicos, e o terceiro é através de exames em DNA. Além de ser a técnica de maior complexidade, o exame em DNA também é a técnica mais demorada. Por esse motivo, uma ferramenta que acelere a identificação das vítimas através do exame em DNA é de grande utilidade.

A solução apresentada neste trabalho contribuiu significativamente na redução tempo de identificação das vítimas. Apenas para comparação, considerando a identificação de vítimas utilizando apenas redes bayesianas, em um acidente com 1.000 vítimas seriam necessárias 500.500 redes. Se o tempo de execução de cada rede for de 1 minuto e 10 segundos em média para testes com 13 locos (GOMES, 2008), aproximadamente 14 meses de processamento ininterrupto seriam necessários para identificar todas as vítimas.

Com a abordagem proposta, considerando testes em 13 locos com degradação da amostra, seriam necessárias 3 horas para o GENESys montar o arquivo de resultados apontando as vítimas mais prováveis para cada família e aproximadamente 105.663 redes bayesianas seriam necessárias para validar esse resultado. Se o tempo de execução de cada rede for de 1 minuto e 10 segundos em média para testes com 13 locos, aproximadamente 85,6 dias seriam gastos na execução das redes. Ou seja, em aproximadamente 86 dias as 1.000 vítimas estariam identificadas.

Convém ressaltar que neste trabalho as simulações realizadas representaram situações reais, desde pequenos acidentes envolvendo algumas dezenas de vítimas até acidentes maiores com milhares de vítimas. As simulações também levaram em consideração os efeitos da degradação da amostra coletada devido à exposição das vítimas ao ambiente ou a ação de outros agentes, como o fogo e a água. Em todos os testes realizados também foram avaliados os efeitos da variação do número de locos testados nos resultados. Como esperado, o aumento no número de locos, aumenta a precisão dos resultados.

Do ponto de vista tecnológico, este trabalho contribui com a utilização das técnicas de verificação simbólica para validar sistemas biológicos. Ressalta-se que são poucas as abordagens para se projetar sistemas biológicos com o suporte da verificação simbólica de modelos. A grande maioria dos trabalhos está centrada na validação de grandes bases de dados biológicas (RACUNAS; SHAH; FEDOROFF, 2006) ou verificação de redes bioquímicas (CHABRIER; FAGES, 2003).

## 6.2 Trabalhos Futuros

Durante o desenvolvimento desta pesquisa, foram identificadas algumas questões que podem ser consideradas à fim de dar continuidade a este trabalho. Algumas sugestões de trabalhos futuros são:

### *6.2.1 Integração com o sistema PedExpert*

A criação e execução das redes bayesianas é um passo importante no processo de identificação de vítimas proposto. Uma das sugestões é a integração do arquivo de resultados gerado pelo GENESys com o sistema PedExpert (GOMES; CAMPOS; PENA, 2009; GOMES, 2008), um sistema especialista bayesiano para análise de casos complexos de determinação de vínculo genético.

Para a integração, o arquivo de resultados gerado pelo GENESys deve ser alterado para um formato que seja compatível com o sistema PedExpert e deve conter informações suficientes para a criação e execução da rede bayesiana sem a necessidade de intervenções manuais. Um arquivo com os resultados da execução das redes pode ser montado pelo PedExpert apresentando o resultado final da validação das vítimas.

### 6.2.2 Ferramenta para reunir amostras em grupos

Outro trabalho interessante, que complementa este, é a criação de uma ferramenta capaz de reunir as amostras de DNA coletadas no local do desastre em grupos com o mesmo perfil genético, cada grupo representando uma vítima. Isso reduzirá o problema de confrontar $M$ amostras a $N$ famílias a um problema menor: confrontar $J$ grupos de amostras a $N$ famílias, onde $J$ é um número muito menor que $M$.

### 6.2.3 Análises estatísticas incorporadas ao GENESys

O processo de identificação de vítimas proposto neste trabalho não elimina a utilização de redes bayesianas para validação dos resultados obtidos no GENESys. Uma solução de verificação que dispense a confirmação da identidade genética através da criação de redes bayesianas, agregando ao processo de verificação análises estatísticas se apresenta como um trabalho a ser desenvolvido. Isso pode ser conseguido através da utilização de verificadores que são capazes de embutir probabilidades nas transições de estado, como por exemplo o verificador PRISM (PARKER, 2002; KWIATKOWSKA; NORMAN; PARKER, 2002; KWIATKOWSKA; NORMAN; PARKER, 2004).

### 6.2.4 Considerar Mutação genética

Um problema que torna mais complexo a determinação da identidade genética é a ocorrência de mutações no DNA dos indivíduos envolvidos. Este trabalho não considera os efeitos da mutação genética no processo de verificação. Uma possível ocorrência de mutação pode gerar no arquivo de resultados do GENESys uma família sem nenhuma vítima com 100% de compatibilidade.

Um trabalho futuro seria o estudo dos efeitos da mutação genética e a adaptação do GENESys para esses casos.

# REFERÊNCIAS


ALBERTS, B. et al. *Biologia molecular da célula*. Porto Alegre: Artmed, 2004.

ALONSO, A. et al. Challenges of DNA Profiling in Mass Disaster Investigations. *Croatian Medical Journal*, v. 46, n. 2, p. 540–548, 2005.

BELLINI, P.; MATTOLINI, R.; NESI, P. Temporal logics for real-time system specification. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 32, n. 1, p. 12–42, 2000. ISSN 0360-0300.

BORGES-OSóRIO, M. R.; ROBINSON, W. M. *Genética humana*. Porto Alegre: Artmed, 2001.

BRYANT, R. E. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, v. 35, p. 677–691, 1986.

BUDOWLE, B.; BIEBER, F. R.; EISENBERG, A. J. Forensic aspects of mass disasters: Strategic considerations for DNA-based human identification. *Legal Medicine*, Elsevier, v. 7, n. 4, p. 230–243, 2005.

CAMPOS, S. et al. Computing Quantitative Characteristics of Finite-state Real-time Systems. In: *In IEEE Real-Time Systems Symposium*. [S.l.]: IEEE, 1994. p. 266–270.

CAMPOS, S. et al. Timing analysis of industrial real-time systems. In: *In Workshop on Industrial-strength Formal specification Techniques*. [S.l.]: IEEE Press, 1995. p. 97–107.

CHABRIER, N.; FAGES, F. Symbolic Model Checking of Biochemical Networks. In: *CMSB 03: Proceedings of the 1st International Workshop on Computational Methods in Systems Biology*. [S.l.]: Springer-Verlag, 2003. p. 149–162.

CLARKE, E. M. et al. Progress on the State Explosion Problem in Model Checking. In: *Informatics - 10 Years Back. 10 Years Ahead.* London, UK: Springer-Verlag, 2001. p. 176–194. ISBN 3-540-41635-8.

CLARKE, E. M.; GRUMBERG, O.; PELED, D. A. *Model Checking*. Cambridge, Massachusetts: The MIT Press, 1999.

CLARKE, E. M.; WING, J. M.; AL, E. Formal methods: State of the art and future directions. *ACM Computing Surveys*, v. 28, p. 626–643, 1996.

DAWID, A.; MORTERA, J.; VICARD, P. Representing and solving complex DNA identification cases using Bayesian networks. *International Congress Series*, v. 1288, p. 484–491, 2006.

GOMES, R. R. *Proposta de um sistema especialista bayesiano para análise casos complexos de determinação de vínculo genético*. Tese (Doutorado) — Universidade Federal de Minas Gerais, Belo Horizonte, MG, 2008.

GOMES, R. R.; CAMPOS, S. V. A.; PENA, S. D. J. PedExpert: a computer program for the application of Bayesian networks to human paternity testing. *Genetics and Molecular Research*, v. 8, p. 273–283, 2009.

GOMES, R. R. et al. Large Scale Genetic Identity Inference Using Symbolic Model Checking. In: *Fifth International Workshop on Constraints in Formal Verification*. Sydney: [s.n.], 2008.

HOFFEE, P. A. *Genética médica molecular*. Rio de Janeiro: Guanabara Koogan, 2000.

IBGE. *A família brasileira*. 2001. Disponível em: `http://www.ibge.gov.br/ibgeteen/pesquisas/familia.html`. Acesso em: 16 mar. 2009.

KWIATKOWSKA, M.; NORMAN, G.; PARKER, D. PRISM: Probabilistic Symbolic Model Checker. *Computer Performance Evaluation: Modelling Techniques and Tools*, Springer Berlin / Heidelberg, p. 113–140, 2002.

KWIATKOWSKA, M.; NORMAN, G.; PARKER, D. Probabilistic symbolic model checking with PRISM: a hybrid approach. *Int. J. Softw. Tools Technol. Transf.*, Springer-Verlag, Berlin, Heidelberg, v. 6, n. 2, p. 128–142, 2004. ISSN 1433-2779.

LANGMEAD, C.; JHA, S. Predicting Protein Folding Kinetics Via Temporal Logic Model Checking. In: *7th International Workshop on Algorithms in Bioinformatics (WABI'07)*. [S.l.]: Springer, 2007. (LNCS, v. 4645), p. 252–264.

LARSEN, P. G.; LASSEN, P. B. The IFAD VDM-SL Toolbox: A Practical Approach to Formal Specifications. *ACM SIGPLAN Notices*, v. 29, p. 77–80, 1994.

LAU, G.; TAN, W. F.; TAN, P. H. After the Indian Ocean Tsunami: Singapore's contribution to the international disaster victim identification effort in Thailand. *Ann Acad Med Singapore*, v. 34, n. 5, p. 341–351, June 2005.

LIN, T.-h.; MYERS, E. W.; XING, E. P. Interpreting anonymous DNA samples from mass disasters—probabilistic forensic inference using genetic markers. *Bioinformatics*, Oxford University Press, Oxford, UK, v. 22, n. 14, p. e298–e306, 2006. ISSN 1367-4803.

MCMILLAN, K. L. *The SMV System DRAFT*. Carnegie Mellon University, Pittsburgh, PA: [s.n.], 1992.

NETO, P. L. de O. C. *Estatística*. São Paulo: E. Blücher, 2002.

NUSSBAUM, R. L.; MCINNES, R. R.; WILLARD, H. F. *Thompson & Thompson genética médica*. Rio de Janeiro: Guanabara Koogan, 2002.

NYMAG. New York Magazine. *9/11 by the Numbers*. 2002. Disponível em: `http://nymag.com/news/articles/wtc/1year/numbers.htm`. Acesso em: 20 out. 2008.

PARKER, D. *Implementation of Symbolic Model Checking for Probabilistic Systems*. Tese (Doutorado) — University of Birmingham, 2002.

PENA, S. D. J. Segurança pública: determinação de identidade genética pelo DNA. *Parcerias Estratégicas*, v. 20, p. 447–460, 2005.

PENA, S. D. J. Thomas Bayes: o cara! *Ciência Hoje*, v. 38, p. 22–29, 2006.

PENA, S. D. J.; PRADO, V. F.; EPPLEN, J. T. DNA diagnosis of human genetic individuality. *Journal of Molecular Medicine*, v. 73, n. 11, p. 555–564, 1995.

RACUNAS, S. A.; SHAH, N. H.; FEDOROFF, N. V. A case study in pathway knowledgebase verification. *BMC Bioinformatics*, v. 7, p. 196, 2006.

SNUSTAD, D. P.; SIMMONS, M. J. *Fundamentos de genética.* Rio de Janeiro: Guanabara Koogan, 2001.

SPIVEY, J. M. *Understanding Z: a specification language and its formal semantics.* New York, NY, USA: Cambridge University Press, 1988. ISBN 0-521-33429-2.

VALMARI, A. The State Explosion Problem. In: *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the volumes are based on the Advanced Course on Petri Nets.* London, UK: Springer-Verlag, 1998. p. 429–528. ISBN 3-540-65306-6.

WALSH, S. J. Recent advances in forensic genetics. *Expert Review of Molecular Diagnostics*, Future Drugs, v. 4, n. 1, p. 31–40, 2004.

WATSON, J. D.; CRICK, F. H. Genetical implications of the structure of deoxyribonucleic acid. *Nature*, v. 171, n. 4361, p. 964–967, 1953.

YANG, J. et al. Using model checking to find serious file system errors. *ACM Trans. Comput. Syst.*, ACM, New York, NY, USA, v. 24, n. 4, p. 393–423, 2006. ISSN 0734-2071.

# APÊNDICE A – MODELO SMV

Aqui está o código completo do modelo na linguagem SMV para o loco CSF1PO de uma família. A modelagem para os demais locos desta família será semelhante, variando apenas nos valores da enumeração (possíveis alelos para o loco).

```
MODULE Mendel (P1, P2, M1, M2, F1, F2)
TRANS
( ((F1=P1 | F1=P2) & (F2=M1 | F2=M2)) |
  ((F1=M1 | F1=M2) & (F2=P1 | F2=P2)) )


MODULE Familia (I1_1, I1_2, I2_1, I2_2,
                II1_1, II1_2, II2_1, II2_2,
                III1_1, III1_2, III2_1, III2_2)
VAR
R1: Mendel (I1_1, I1_2, I2_1, I2_2, II1_1, II1_2);
R2: Mendel (II1_1, II1_2, II2_1, II2_2, III1_1, III1_2);
R3: Mendel (II1_1, II1_2, II2_1, II2_2, III2_1, III2_2);


MODULE main
VAR
I2_1 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
I2_2 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
II1_1 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
II1_2 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
III2_1 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
III2_2 : {7, 8, 9, 10, 11, 12, 13, 14, 15};
```

```
L1 : Familia (9,12, I2_1,I2_2, II1_1,II1_2, 12,15, 12,12, III2_1,III2_2);

SPEC EF ( III2_1 = 12 & III2_2 = 7 )
SPEC EF ( III2_1 = 14 & III2_2 = 14 )
SPEC EF ( III2_1 = 10 & III2_2 = 11 )
SPEC EF ( III2_1 = 12 & III2_2 = 10 )
SPEC EF ( III2_1 = 12 & III2_2 = 11 )
SPEC EF ( III2_1 = 11 & III2_2 = 12 )
SPEC EF ( III2_1 = 12 & III2_2 = 12 )
SPEC EF ( III2_1 = 10 & III2_2 = 12 )
SPEC EF ( III2_1 = 10 & III2_2 = 13 )
SPEC EF ( III2_1 = 10 & III2_2 = 9 )
SPEC EF ( III2_1 = 13 & III2_2 = 13 )
SPEC EF ( III2_1 = 10 & III2_2 = 8 )
SPEC EF ( III2_1 = 10 & III2_2 = 12 )
SPEC EF ( III2_1 = 13 & III2_2 = 12 )
SPEC EF ( III2_1 = 10 & III2_2 = 9 )
SPEC EF ( III2_1 = 13 & III2_2 = 13 )
SPEC EF ( III2_1 = 10 & III2_2 = 7 )
SPEC EF ( III2_1 = 12 & III2_2 = 10 )
SPEC EF ( III2_1 = 12 & III2_2 = 11 )
SPEC EF ( III2_1 = 14 & III2_2 = 11 )
```

# Livros Grátis

( http://www.livrosgratis.com.br )

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo