UNIVERSIDADE FEDERAL DO CEARÁ
DEPARTAMENTO DE COMPUTAÇÃO
MESTRADO E DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

DIEGO VICTOR SIMÕES DE SOUSA

REALIZANDO CONSULTAS EM TRAJETÓRIAS SEMÂNTICAS UTILIZANDO UMA ABORDAGEM BASEADA EM VERIFICAÇÃO DE MODELOS

FORTALEZA, CEARÁ

2012

DIEGO VICTOR SIMÕES DE SOUSA

REALIZANDO CONSULTAS EM TRAJETÓRIAS SEMÂNTICAS UTILIZANDO UMA ABORDAGEM BASEADA EM VERIFICAÇÃO DE MODELOS

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Área de concentração: Banco de Dados

Orientador: Prof. Dr. José Antonio Fernandes de Macêdo

FORTALEZA, CEARÁ

2012

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca de Ciências e Tecnologia

S696r Sousa, Diego Victor Simões de.
Realizando consultas em trajetórias semânticas utilizando um a bordagem baseada em verificação de modelos. / Diego Victor Simões de Sousa. – 2012.
80f. : il. color., enc. ; 30 cm.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Ciências, Departamento de Computação, Programa de Pós-Graduação em Ciência da Computação, Fortaleza, 2011.
Área de Concentração: Ciência da Computação.
Orientação: Prof. Dr. José Antonio Fernandes de Macêdo.

1. Semântica – análise de redes. 2. Sistema de posicionamento global. 3. GPS. 4. Sistema de comunicação móvel. 5. Sistemas de localização automática de veículo a motor. I. Título.

CDD 005

DIEGO VICTOR SIMÕES DE SOUSA

REALIZANDO CONSULTAS EM TRAJETÓRIAS SEMÂNTICAS UTILIZANDO UMA ABORDAGEM BASEADA EM VERIFICAÇÃO DE MODELOS

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação, da Universidade Federal do Ceará, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação. Área de concentração: Banco de Dados

Aprovada em: __/__/____

BANCA EXAMINADORA

Prof. Dr. José Antonio Fernandes de Macêdo
Universidade Federal do Ceará - UFC
Orientador

Prof. Dr. Javam de Castro Machado
Universidade Federal do Ceará - UFC

Profa. Dra. Vânia Maria Ponte Vidal
Universidade Federal do Ceará - UFC

Prof. Dr. Marco Antonio Casanova
Pontifícia Universidade Católica do Rio de Janeiro - PUC-Rio

À Deus. Aos meus Pais.

AGRADECIMENTOS

Agradeço, primeiramente, a DEUS pela vida que tenho e pela oportunidade que me foi dada ao longo da vida para desempenhar tamanho trabalho.

Agradeço aos meus familiares (Vicente, Sandra, Igor e Iury), principalmente aos meus pais, Vicente e Sandra, que sempre me incentivaram nos estudos. Sem as palavras de força nos momentos de vitórias e nos momentos de derrotas, e sem os ensinamentos sobre a vida, muitas vezes mais importantes do que os conhecimentos teóricos, não teria chegado tão longe. Sou eternamente grato a eles por minha criação.

Agradeço a minha noiva Nathália, que durante o mestrado sempre esteve ao meu lado e desempenhou um companheirismo tremendo, compreendendo quando eu não podia estar tão presente, devido ao tempo de estudos despendido. No decorrer do mestrado ela era minha namorada, ao término dele, foi recompensada com o início do noivado.

Aos grandes amigos que fiz no mestrado, alguns destes me ajudaram com relação às disciplinas ou com dicas relacionadas ao trabalho de dissertação, dentre eles, pode-se citar: Wagner, Francicleber, Flávio, David, Jeovane Reges, Arlino, Vinícius, Paulo (Doido), Hélio, Danusa, Fernanda Lígia, Ticianne, Bruno Leal, Lívia, Ticiana, Macêdo, Cleiton e Fernando Wagner. Espero não ter me esquecido de ninguém.

Aos amigos especiais que fiz e que fizeram uma grande diferença em minha vida. À Mônica Regina, que me ajudou bastante durante as disciplinas, além de ter participado de discursões diversas comigo, as quais proporcionaram o amadurecimento da mente. Ao Regis Pires, que se mostrou um amigo para toda hora, além de proporcionar muitos ensinamentos sobre diversos assuntos relacionados com Tecnologia da Informação. Ao Igo Brilhante, que me ajudou bastante com algumas ferramentas, além de me fornecer os dados para a realização dos testes. Ao Diego Sá, que sempre me incentivou quando estava triste e cabisbaixo. Ao Manoel Siqueira, com quem dividi um apartamento em Redenção, nos momentos finais da dissertação. O Manoel se mostrou um grande amigo. Nos momentos finais, ele me incentivava quando estava desmotivado por algum motivo, de modo semelhante, eu o motivava quando ele estava desmotivado. Sem este grande amigo, não sei se teria conseguido concluir o presente trabalho com a mesma galhardia.

Ao grande amigo de mestrado Henrique Viana, que teve muita paciência para me ajudar a entender um pouco sobre lógica temporal. O Henrique se mostrou um verdadeiro professor, além de ter me ajudado algumas vezes nas disciplinas e ter me incentivado em diversos momentos. Obrigado Henrique!

Ao professor José Fernandes Antonio de Macêdo, que me abriu a oportunidade de trabalhar juntamente com ele no tema da presente dissertação. O professor Tonho, como é conhecido, se mostrou muito paciente, principalmente nos momentos em que eu me angustiava com alguns problemas, que foram superados ao longo da dissertação. Nas nossas conversas, discutimos algumas vezes, mas sempre conseguíamos nos conciliar, visando o objetivo maior, que era a conclusão do trabalho. O professor Tonho se mostrou um amigo em diversos momentos,

sou muito grato a ele.

Ao professor Davi Romero, da UFC Quixadá, que proporcionou algumas discussões acerca de Lógica Temporal, ajudando na compreensão do referido tema.

Ao pesquisador Nicolas Markey, que mesmo à distância (França), me ajudou bastante a entender alguns conceitos referentes ao desenvolvimento do presente trabalho. Sou muito grato às ajudas por ele fornecidas.

À UNILAB, *Universidade da Integração Internacional da Lusofonia Afro-Brasileira*, instituição da qual passei a fazer parte de seu quadro de servidores ao final de mestrado. O apoio desta instituição foi imprescindível para o sucesso do presente trabalho, já que foram viabilizados espaço físico, internet e computadores para que eu pudesse ficar pesquisando e escrevendo a dissertação além do horário do expediente. Agradeço muito à Pró-Reitora de Administração, Adênia, por ter viabilizado a minha presença na universidade em horários especiais.

Agradeço aos meus companheiros de trabalho da CTI, *Coordenação de Tecnologia da Informação*, por toda a compreensão durante a conclusão desta dissertação, principalmente aos meus chefes, Cadu e Ladislav, que viabilizaram horários flexíveis de trabalho para que eu conseguisse concluir o presente trabalho. Não posso esquecer de citar os demais companheiros de CTI, Thiago, Débora, Marcos e Manoel Siqueira. Mais uma vez agradeço ao Manoel, com quem dividi alguns momentos difíceis e alegres durante as noites na UNILAB, em Redenção.

Agradeço também ao Pró-Reitor de Planejamento da UNILAB, professor Fernando Afonso, que deu todo o apoio a mim e ao Manoel para que concluíssemos o Mestrado, demonstrando que a Pró-Reitoria de Planejamento, a qual a CTI é subordinada, estava o tempo todo apoiando a qualificação de seus funcionários.

Ao professor Marco Antonio Casanova, pela participação em minha banca de mestrado.

À professora Bernadette Farias Lóscio, pela oportunidade que me foi dada de ingressar no Mestrado em Computação. Serei eternamente grato.

À galera estourada da Computação UFC: Franzé, Victor, Toni e Camila.

À FUNCAP (Fundação Cearense de Apoio a Pesquisa), pelo apoio financeiro durante todo o mestrado.

A todos os professores e, principalmente, alunos, membros do grupo ARIDA (Advanced Research In DAtabase), pelo os esforços desempenhados visando à construção de um grupo de pesquisa de ponta em banco de dados.

A todos aqueles que contribuíram, de alguma forma, tanto no lado do trabalho propriamente dito, quanto no lado do apoio psicológico, que eu possa ter esquecido.

“A genialidade é 1% inspiração e 99% transpiração.”

(Thomas Edison)

RESUMO

A popularização de dispositivos móveis equipados com serviços de localização geográfica (e.g. GPS) está permitindo a coleta de dados de trajetórias de objetos móveis de forma rápida e barata. O armazenamento destes dados vem possibilitando o desenvolvimento de novos tipos de aplicações que podem utilizar esses dados para realizar análise sobre o comportamento de objetos móveis. Porém, realizar tais análises a partir de dados brutos, gerados pelo dispositivo de localização geográfica, é um grande desafio, visto que tais dispositivos apenas coletam as informações sobre as coordenadas e o instante de tempo, por onde o objeto móvel se deslocou. Apesar dos diversos esforços empreendidos na busca de soluções para enriquecer dados de trajetórias de objetos móveis com informações semânticas da aplicação, pouco foi realizado no sentido de prover mecanismos para consultar tais trajetórias enriquecidas. Foi percebido, então, que a falta de métodos para processamento de consultas sobre trajetórias semânticas, em especial, consulta sobre padrões de movimento, são um obstáculo para a realização de análises de interesse de uma grande parte das aplicações deste domínio. Desta forma, as principais contribuições deste trabalho são: (1) um método para processar e realizar consultas que descrevem um padrão de movimento constituído de uma sequência de conjuntos de predicados que podem ocorrer em uma trajetória semântica armazenada em um banco de dados e (2) definição de uma linguagem para expressar padrões de movimento sobre trajetórias semânticas. Com objetivo de validar a proposta apresentada, desenvolvemos um sistema que permite a especificação de uma consulta para expressar um padrão de movimento na linguagem definida. Além disso, utilizamos tal ferramenta para realizar testes sobre um banco de dados de trajetórias de carros da cidade de Milão, semanticamente enriquecidas com informações do aplicativo *Foursquare*. Os resultados obtidos mostraram que a complexidade para processar as consultas é linear com relação ao número de trajetórias semânticas e o número de predicados na consulta, considerando poucas trajetórias com muitos episódios e muitas trajetórias com poucos episódios. A abordagem proposta superou outras abordagens existentes, tanto no que concerne a performance do processamento de consultas, quanto na expressividade das consultas que podem ser escritas na linguagem proposta.

Palavras-chave: trajetória semântica. consultas de correspondência de padrões. expressão regular.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 Motivação

A popularização de dispositivos móveis equipados com serviços de localização geográfica (e.g. GPS) está permitindo a coleta de dados de trajetórias de objetos móveis de forma rápida e barata. O armazenamento destes dados vem possibilitando o desenvolvimento de novos tipos de aplicações que podem utilizar esses dados para realizar análise sobre o comportamento de objetos móveis. Porém, realizar tais análises a partir de dados brutos, gerados pelo dispositivo de localização geográfica, é um grande desafio, dado o volume de dados, a falta de informações semânticas sobre as trajetórias e a necessidade de realizar descoberta de padrões associados com a movimentação de objetos móveis. Desta forma, desenvolver métodos e técnicas que permitam superar tais desafios é chave para permitir o desenvolvimento de aplicações que visem descobrir ou detectar o comportamento de objetos móveis a partir de suas trajetórias.

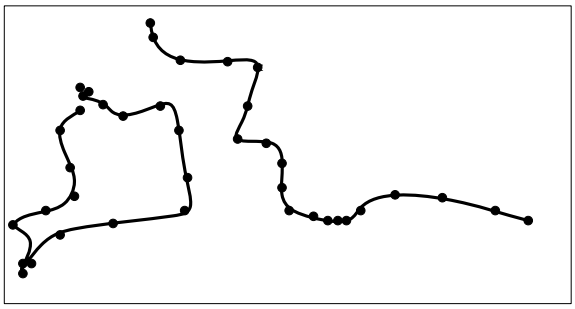
Alguns exemplos de aplicações que fazem uso de trajetórias são: gerenciamento de trânsito, migração de pássaros, gestão de movimentação de conjunto de pessoas, planejamento de rotas de veículos de entrega, citando apenas algumas. Algo comum em todas essas aplicações é a necessidade de se verificar a ocorrência de padrões de comportamento a partir da análise dos dados de trajetória. Este padrão de comportamento pode ser descrito por uma sequência de eventos que ocorrem durante um determinado período. Por exemplo, no caso de uma aplicação de migração de pássaros, um biólogo poderia estar interessado em identificar os pássaros que possuem o seguinte padrão de comportamento: *pássaros que saem da Europa no outono em direção à Franca, atravessam o Mediterrâneo, acasalam nas proximidades de uma região com água, realizam paradas de no máximo 3 dias, e, após 4 dias, finalizam sua migração na África do Sul*. Neste exemplo, o padrão de comportamento é descrito por uma sequência de informações semânticas, com restrições temporais (“no máximo 3 dias”), que podem representar um predicado espacial (“saem da Europa”), uma descrição de um local visitado (“África do Sul”) ou uma atividade realizada (“acasalam”).

Entretanto, um dispositivo de localização geográfica geralmente captura dados relacionados com a movimentação de objetos móveis através de um conjunto de *pontos de localização*, descrito por triplas contento: tempo, latitude e longitude. Estes pontos, apesar de identificarem o posicionamento de um *objeto móvel* ao longo do tempo, carecem de informação semântica, o que dificulta a busca por trajetórias similares a um dado padrão de comportamento. Desta forma, a descoberta de padrões associados com a movimentação do objeto móvel depende do processo de enriquecimento semântico das trajetórias brutas coletadas pelos dispositivos de localização geográfica.

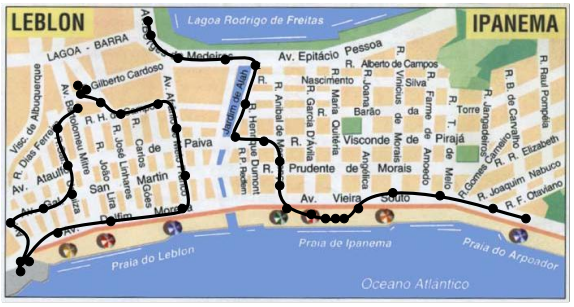
Visando facilitar a análise de trajetórias, diversos trabalhos foram desenvolvidos visando associar informações semânticas aos pontos de localização de uma trajetória. Denominamos o processo de associar informações aos pontos espaço-temporais das trajetórias de processo de enriquecimento semântico de trajetórias. Este processo está diretamente ligado ao domínio da aplicação. Por exemplo, no cenário acima, as atividades que devem ser associadas

aos pássaros são aquelas que estão relacionadas com o comportamento de pássaros definidos pelo domínio da aplicação (e.g. Biologia). Sendo assim, faz-se necessário o uso da ideia de trajetória semântica introduzida por (SPACCAPIETRA et al., 2008). Estas trajetórias podem ser interpretadas como uma sequência de informações semânticas, também chamadas de anotações (e.g. trabalhando, andando de bicicleta), que são válidas durante um intervalo de tempo para uma dada aplicação.

A importância do processo de enriquecimento semântico pode ser mais bem compreendida através da Figura 1.1[1] são expostos exemplos de representações de trajetórias na forma bruta, com informações geográficas sobre o domínio de aplicação e com informações semânticas acerca da trajetória e do domínio. Pode-se notar que, para o primeiro caso (Figura 1.1a), a realização de consultas sobre padrão de comportamento do objeto móvel pode ser inviável, devida a ausência de informação semântica sobre a trajetória. Para o segundo caso (Figura 1.1b), a realização de consultas a padrões de movimento será possível, no entanto será necessário um processamento para inferir os locais visitados, assim como as atividades realizadas. Com relação ao último caso (Figura 1.1c), a realização de consultas será facilitada devido às trajetórias serem expressas por uma sequência de eventos que podem se repetir diversas vezes ao longo do trajeto realizado por um objeto móvel. Claramente, neste último caso será necessário inferir as atividades, pontos de interesse, meios de transporte, e outras informações semânticas baseando-se nos dados da trajetória bruta.



(a) Trajetórias vistas como uma sequência de pontos de localização



(b) Trajetórias, juntamente com a informação geográfica do domínio de aplicação

(c) Trajetórias enriquecidas semanticamente

Figura 1.1: Representações de trajetórias

De acordo com o exemplo acima, o enriquecimento semântico de trajetórias dá suporte a realização de consultas que utilizam a semântica da aplicação na especificação de um determinado padrão de movimento. Entretanto, diante de um conjunto de trajetórias semânti-

[1]Figuras construídas a partir de imagens obtidas em http://www.clker.com/

cas, expressar consultas que descrevam um padrão de movimento com restrições temporais é ainda uma questão em aberto na pesquisa desta área. Isto ocorre devido às restrições temporais e à restrição sequencial no qual os eventos devem ser verificados na trajetória. No padrão SQL atual, por exemplo, não é permitida a realização de consultas envolvendo sequência temporal entre as tuplas baseada no valor de um dado atributo. Desta forma, existe a necessidade uma linguagem de consulta que permita expressar consultas sobre trajetórias semânticas, as quais envolvam uma sequência de eventos que possuam restrições temporais. Além disso, é necessário o desenvolvimento de métodos eficientes para processamento dessas consultas sobre um grande volume de dados de trajetórias semânticas.

Alguns trabalhos, na literatura, visam a realizações de consultas sobre trajetórias, buscando aquelas que correspondem a um determinado padrão descrito na consulta. Entretanto, nenhum dos trabalhos analisados utilizou trajetórias semânticas como objetos de primeira classe, ou seja, não utilizou dados que representassem uma trajetória semanticamente enriquecida. Em alguns casos, trabalhos anteriores conseguem realizar consultas envolvendo alguma semântica, entretanto, para que isto seja possibilitado, eles utilizam correlações entre trajetórias representadas de forma bruta e relações que possuem informações acerca do domínio da aplicação. Portanto, existe uma carência de soluções que utilizem a representação semântica de trajetórias.

Motivado por trabalhos que visam o enriquecimento semântico de trajetórias e pela dificuldade em processar consultas sobre trajetórias semânticas que correspondam a um determinado padrão, este trabalho pretende atacar o **problema de expressar consultas sobre padrões de movimento e processar tais consultas no sentido de recuperar trajetórias que correspondam a tais padrões**, considerando que as trajetórias já estejam semanticamente enriquecidas. Para a solução deste problema, o trabalho apresenta um modelo de dados para a representação de trajetórias semânticas baseado em modelos propostos por trabalhos anteriores. Este modelo servirá de base para a definição de uma linguagem de consultas que visa possibilitar expressar padrões de movimento a serem buscados nas trajetórias semânticas. Além disso, dado uma consulta expressa na linguagem definida por este trabalho e dado um conjunto de dados de trajetórias semânticas representadas pelo modelo proposto, será apresentado, também, uma proposta de processamento da linguagem que objetiva retornar os resultados esperados. Ao final, experimentos, utilizando a abordagem proposta, foram realizados e os resultados mostraram-se satisfatórios.

1.2 Objetivos

Diante da motivação na seção anterior, o objetivo geral deste trabalho consiste na especificação de uma linguagem de consultas que possibilite especificar padrões de movimento descritos como uma sequência de eventos com restrições temporais, que vise à realização de consultas sobre trajetórias semânticas. Para atingir este objetivo, os seguintes objetivos específicos devem ser alcançados:

- Definição de um modelo de dados que permita representar as trajetórias semânticas em

um banco de dados;

- Definição de uma abordagem para o processamento das consultas de padrões de movimento;
- Especificação e descrição da semântica da linguagem de consultas para padrões de movimento sobre trajetórias semânticas;
- Analisar a eficiência da abordagem do ponto de vista computacional e compará-la com outras abordagens.

1.3 Contribuições

Como contribuições desta dissertação, obtidas a partir do desenvolvimento deste trabalho no sentido de resolver o problema exposto, pode-se citar:

- Uma linguagem de consulta para padrões de movimento sobre trajetórias semânticas. Esta contribuição visa tanto propor a estrutura da linguagem em BNF como descrever a semântica de seus componentes;
- Um método para processamento das consultas de padrões de movimento, de acordo com a linguagem de consultas definida. Neste sentido propomos um método baseado em checagem de modelos (em inglês Model Checking), visando obter uma computação eficiente para execução da consulta;
- Experimentos sobre dados reais, visando analisar a eficiência da abordagem do ponto de vista computacional e compará-la com outras abordagens.

1.4 Publicações

Ao longo da realização deste trabalho, dois artigos foram publicados com partes da solução proposta. No primeiro artigo, as primeiras definições do modelo de dados e da linguagem proposta foram apresentadas. Já no segundo trabalho, parte inicial da abordagem de processamento de consultas foi exposta, juntamente com uma análise de sua complexidade. Os trabalhos publicados são listados abaixo:

- SIMÕES, D.; MACEDO, J. D. Uma linguagem de consulta para padrões espaço-temporais em trajetórias semânticas. *Proceedings of 26th Brazilian Symposium on Databases - Short Paper*, Sociedade Brasileira de Computação, 2011. (SIMÕES; MACEDO, 2011)
- SIMÕES, D.; VIANA, H.; MARKEY, N.; MACEDO, J. D. Querying trajectories through model checking based on timed automata. *Proceedings of 27th Brazilian Symposium on Databases - Short Paper*, Sociedade Brasileira de Computação, p. 33–40, 2012. (SIMÕES et al., 2012)

1.5 Organização

Esta dissertação está organizada da seguinte forma. No Capítulo 2 o problema a ser atacado neste trabalho é caracterizado, incluindo a definição de um modelo de dados para trajetórias semânticas. Em seguida, no Capítulo 3 os trabalhos relacionados são destacados e discutidos. A solução para processamento de consultas é apresentada no Capítulo 4, onde são introduzidos os conceitos de lógica temporal, verificação de modelos e autômatos temporizados. A solução proposta neste capítulo é baseada em verificação de modelos em autômatos temporizados, na qual as consultas serão escritas como fórmulas de uma lógica temporal. Ainda neste capítulo, é apresentada uma linguagem de consultas que simplifica as fórmulas lógicas para estruturas menores, não obrigando ao usuário entender de lógica temporal para escrever uma consulta. No Capítulo 5, são expostos os resultados experimentais a partir da construção de uma ferramenta para consulta de trajetórias semânticas. Finalmente, no Capítulo 6, é apresentada a conclusão deste trabalho, resumindo os resultados desta dissertação e apresentando os trabalhos futuros.

2 CARACTERIZAÇÃO DO PROBLEMA

O foco deste trabalho é o problema de se recuperar trajetórias semânticas que correspondam a um dado padrão de movimento. Como forma de proporcionar um maior entendimento deste problema, fundamentos conceituais a respeito de dados móveis e trajetórias são apresentados na Seção 2.1. Após isto, a definição do problema é apresentada com mais detalhes na Seção 2.2.

2.1 Preliminares

O estudo de dados móveis consiste em analisar o movimento realizado por entidades, que são geralmente chamadas de objetos móveis. Os objetos móveis se caracterizam por ser algum meio de transporte (e.g. ônibus, bicicleta, avião, etc.), ou um indivíduo equipado com um dispositivo de localização geográfica (e.g. GPS) que desenvolvem algum tipo de movimento. Dados meteorológicos, como tempestades, capturados por satélites também podem ser vistos como objetos móveis. Os movimentos dos objetos móveis são representados por dados espaço-temporais, estes que constituem trajetórias. Uma trajetória representa o trajeto de um objeto móvel, na qual a parte espacial indica a posição do objeto sobre a superfície da terra e a parte temporal identifica o instante de tempo no qual o objeto móvel estará nesta determinada posição.

Do ponto de vista conceitual, uma trajetória é, por definição, um conceito espaço-temporal que caracteriza a evolução da posição de um objeto móvel ao longo do tempo. O deslocamento do objeto móvel no espaço visa atingir uma meta levando uma quantidade específica de tempo, portanto, uma trajetória é definida por um intervalo de tempo. Este intervalo de tempo é limitado pelo instante inicial que um objeto inicia seu trajeto (t_{begin}) e pelo instante de tempo que o objeto finaliza seu trajeto (t_{end}). Desta forma, o intervalo de tempo no qual caracterizará um movimento de um determinado objeto estará compreendido por t_{begin} e t_{end}. Assim, (SPACCAPIETRA et al., 2008) define uma trajetória como segue:

Definição 2.1.1 (Conceito de Trajetória) *"Uma trajetória é o registro, definido pelo usuário, da evolução da posição (percebida como um ponto) de um objeto que está se movimentando no espaço, durante um dado intervalo de tempo, objetivando atingir uma determinada meta." (SPACCAPIETRA et al., 2008)*
Trajetória: $[t_{begin}, t_{end}] \longrightarrow espaco.$

O elemento básico da trajetória é uma observação espaço-temporal consistindo de uma tripla $(id, location, time)$, onde *id* é o identificador único usado em todos os registros do movimento do indivíduo, $location$ é um descritor espacial (e.g. um par ordenado, um polígono, etc.) e $time$ é o instante de tempo no qual o indivíduo está na mencionada localização (dependendo da aplicação, pode ser em minutos ou até mesmo anos) (SPACCAPIETRA et al., 2008). Uma trajetória bruta, então, pode ser representada como uma sequência de triplas

$\langle traj_{id}, x_i, y_i, t_i \rangle$, tal que $traj_{id}$ é o identificador da trajetória, x_i e y_i representam a posição geográfica do objeto móvel (e.g. latitude e longitude), e t_i representa o instante de tempo para uma tupla i pertencente à sequência. Na Figura 2.1[1] é apresentado um exemplo de trajetória bruta, definida mais adiante de acordo com (ALVARES et al., 2007b), onde o seu identificador ($traj_{id}$) e o instante de tempo (t_i) foram omitidos.

Figura 2.1: Exemplo de uma trajetória bruta. O identificador de cada ponto é o mesmo, apesar de estar ausente, juntamente com o instante de tempo

Definição 2.1.2 (Trajetória Bruta) *Definição do trabalho (ALVARES et al., 2007b). Uma trajetória é uma sequência de pontos espaço-temporais* $\langle (x_0, y_0, t_0), (x_1, y_1, t_1), ..., (x_n, y_n, t_n) \rangle$*, na qual* $x_i, y_i \in \mathbb{R}$ *e* $t_i \in \mathbb{R}^+$*, para* $i = 0, ..., n$ *e* $t_0 < t_1 < \cdots < t_n$*.*

Uma sequência de triplas coletadas por um dispositivo de localização pode, de fato, representar diferentes trajetórias de um determinado objeto móvel. Por exemplo, suponha que um indivíduo trafegue diversas vezes ao dia em uma cidade com um dispositivo GPS. Ao final do dia, a coleção de triplas capturadas pelo dispositivo pode ser separada em diferentes trajetórias, cada uma referente a um deslocamento feito pelo indivíduo. Em outro âmbito, suponha que o mesmo conjunto de triplas seja utilizado por uma aplicação de gerência de tráfego da cidade. Neste segundo cenário, a trajetória do indivíduo deve ser definida como uma sequência de todas as triplas coletadas ao longo do dia. De maneira geral, a definição do que é uma trajetória irá depender da aplicação, ou seja, a definição do início e do final de uma trajetória irá depender de como a aplicação compreende o movimento de um objeto móvel.

Portanto, dependendo da aplicação, o projetista do banco de dados deve definir quais dados brutos, coletados por um dispositivo de posicionamento, serão pertencentes a uma determinada trajetória. Estas serão armazenadas em um banco de dados e manipuladas pela

[1]Figura adquirida através da ferramenta OpenJump: `http://www.openjump.org/index.html`

aplicação. Na literatura, o processo de segmentar dados de trajetórias brutas é conhecido como construção semântica de trajetórias (ALVARES et al., 2007b). Em outro âmbito, um objeto móvel não estará, necessariamente, sempre em movimento. As trajetórias podem ser segmentadas em subintervalos de tempo onde a posição muda constantemente ou onde a posição permanece a mesma, caracterizando uma trajetória como uma sequência de movimentos, indo de uma parada para outra. Esta sequência de movimentos entre paradas é o que comumente realizamos no nosso trajeto ao longo do dia. Por exemplo, um turista pode, ao aterrizar no aeroporto de uma determinada cidade, se dirigir a um local turístico, ficar um tempo neste local tirando fotos, e depois ir ao seu hotel, caracterizando, assim, sua trajetória como uma sequência de locais visitados durante um intervalo de tempo.

Seguindo este raciocínio, em (SPACCAPIETRA et al., 2008), o conceito de trajetória semântica é definido de acordo com um modelo conceitual baseado em episódios de paradas e de movimentos (i.e. *stops* e *moves*), no qual as paradas representam pontos espaço-temporais, onde ocorreu pouca ou nenhuma movimentação ao longo do tempo, e os movimentos representam o período no qual o objeto móvel estava em movimento. Desta forma, as trajetórias semânticas podem ser representadas por uma sequência de movimentos indo de uma parada para a próxima (SPACCAPIETRA et al., 2008). A Figura 2.2 exemplifica uma trajetória vista como uma sequência de pontos visitados ao longo do tempo e como uma trajetória semântica definida por uma sequência de episódios.

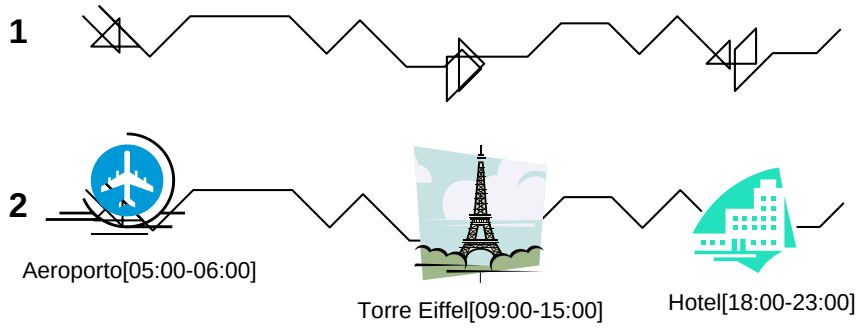


Figura 2.2: (1) Uma trajetória bruta e (2) a mesma trajetória vista como uma trajetória semântica

A fim de esclarecer a diferença entre a representação de trajetórias com e sem informação semântica acerca do domínio de aplicação, o trabalho (ALVARES et al., 2007b) apresenta a Figura 2.3. Nesta figura é apresentada uma representação de trajetórias sem informação semântica (esquerda) e com informações semânticas acerca do domínio da aplicação, onde se pode inferir que a região representa uma parte de Paris, onde um dos locais representados é a Torre Eiffel. Desta forma, uma trajetória semântica não seria mais vista como uma sequência de triplas (x, y, t), mas como uma sequência de locais visitados durante um determinado intervalo de tempo.

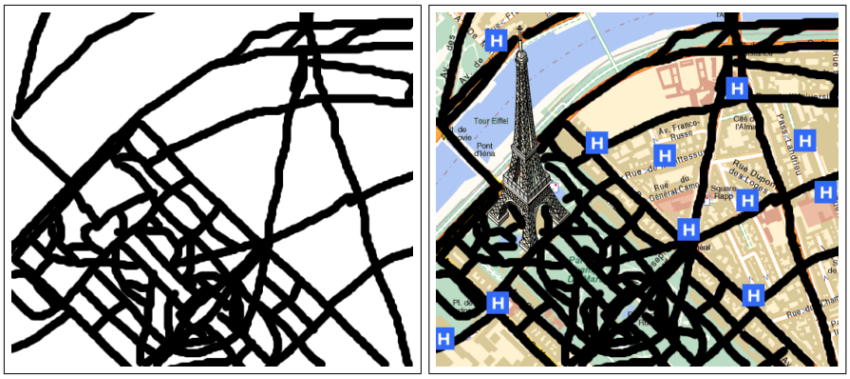


Figura 2.3: (À esquerda) trajetórias brutas e (à direita) trajetórias brutas juntamente com a informação geográfica (ALVARES et al., 2007a)

Seguindo trabalhos que enfatizam a necessidade de abordar a análise de trajetórias através do ponto de vista semântico, (ALVARES et al., 2007b; SPACCAPIETRA et al., 2008; CAO; CONG; JENSEN, 2010), alguns trabalhos propõem métodos que objetivam construir trajetórias semânticas a partir de trajetórias representadas por uma sequência de pontos de localização (BAGLIONI et al., 2008; XIE; DENG; ZHOU, 2009; YAN et al., 2010, 2011). Em (YAN et al., 2010), por exemplo, é proposto um modelo semântico-espacial de trajetórias para construção de diferentes níveis de abstração semântica de dados brutos capturados por dispositivos GPS (e.g. tuplas (x,y,t)). Neste trabalho uma trajetória semântica é definida como uma sequência de episódios de parada e de movimento, que possuem uma representação geométrica de um local pertencente ao espaço geográfico e um conjunto de anotações semânticas que representam informações válidas para um determinado episódio, por exemplo, uma atividade realizada, a descrição de um local ou o meio de transporte utilizado. Este trabalho utiliza conhecimento, acerca do domínio de aplicação, para gerar informações temáticas (semântica) relacionadas com a aplicação, enriquecendo trajetórias brutas. Abaixo segue a definição formal de trajetória semântica utilizada no trabalho (YAN et al., 2011) e que servirá de base para a definição do problema a ser resolvido nesta dissertação.

Definição 2.1.3 (Trajetória Semântica - Yan et. al 2011) *"Uma trajetória semântica é uma sequência de episódios definidos por um conjunto de predicados" (YAN et al., 2011). Uma trajetória semântica ST é definida como* $ST = \{ep_1, ep_2, \cdots, ep_m\}$*, tal que cada episódio corresponde a uma subsequência da trajetória original e é representado como uma tupla* $ep_i = (sp, time_{in}, time_{out}, A)$*, onde sp é um local semântico definido por uma representação geométrica,* $time_{in}$ *e* $time_{out}$ *são, respectivamente, o tempo que o objeto móvel entra e sai de sp, e A é um conjunto de anotações semânticas associado ao episódio.*

2.2 Definição do Problema

Nesta seção serão definidos os conceitos referentes ao problema abordado nesta dissertação e quais os problemas gerados pela solução proposta.

De um ponto de vista conceitual, uma trajetória pode ser vista por uma aplicação como um objeto identificável e gerenciável. Mesmo que a trajetória seja definida como um objeto de primeira classe, as aplicações, muitas vezes, exigem relacionar informações a segmentos de uma trajetória para, por exemplo, indicar que um indivíduo estava caminhando em um dado intervalo de tempo ao longo da trajetória. Assim, uma visão conceitual de trajetórias deve fornecer mecanismos para lidar com as trajetórias como objetos de primeira classe, descritas como uma decomposição de episódios significativos.

Trabalhos recentes têm proporcionado o enriquecimento semântico de dados de trajetórias no sentido de facilitar a descrição do comportamento dos objetos móveis (e.g. um automóvel, uma pessoa com um dispositivo GPS) (YAN et al., 2010, 2011). Essas abordagens utilizam conhecimento do domínio da aplicação para gerar informações temáticas (semântica) relacionadas com a aplicação, enriquecendo trajetórias brutas.

A premissa do problema, abordado neste trabalho, considera que existam **trajetórias semânticas**, como objetos de primeira classe, armazenadas em um banco de dados. Estas trajetórias são representadas por uma sequência de episódios, seguindo o conceito de (YAN et al., 2011) com algumas diferenças, onde cada episódio (de parada ou de movimento) possui um intervalo de tempo no qual um conjunto de predicados, que podem representar predicados espaço-temporais e/ou informações semânticas, são válidos. Desta forma, uma trajetória semântica é vista como uma sequência de intervalos de tempo (chamados de episódios), que não se sobrepõem, nos quais determinados predicados, provenientes do domínio de aplicação, são válidos.

Definição 2.2.1 (Trajetória Semântica - TS**)** *Seja* Φ *o conjunto de todas as descrições de predicados do domínio de aplicação, uma* ***trajetória semântica*** *é definida pela seguinte tupla* $TS = (id, start, end, EL)$*, tal que id representa o identificador da trajetória,* $start, end \in T \subseteq \mathbb{N}$ *representam, respectivamente, os instantes de tempo inicial e final da trajetória, sendo* T *o conjunto dos instantes de tempo pertencentes a uma dada aplicação, representados, cada um, por um único número inteiro não negativo.* $EL = \{(start_1, end_1, \Psi_1), (start_2, end_2, \Psi_2), \cdots, (start_n, end_n, \Psi_n)\}$ *é uma lista sequencial de episódios, tal que para* $1 \leq i \leq n$*,* $start_i < end_i$*,* $start_{i+1} = end_i$ *e* $\Psi_i \subseteq \Phi$ *ou* $\Psi = \emptyset$ *(ou seja, é um subconjunto do conjunto de todas as descrições de predicados pertencentes à aplicação, que pode ser vazio). Para um episódio* $ep = (start, end, \Psi)$*, start e end representam, respectivamente, o instante de tempo inicial e final do episódio e* $\Psi \subseteq \Phi$ *representa o conjunto de predicados válidos durante o intervalo de tempo* $[start, end]$.

Do ponto de vista conceitual, a representação do modelo de dados baseado na definição de **trajetória semântica** pode ser apresentada em um diagrama UML, como exposto na Figura 2.4.

Podem-se observar na Figura 2.4 que uma **trajetória** é composta por 1 ou mais **episódios**, os quais são específicos para cada trajetória. Cada **episódio** será associado com 1 ou mais **predicados**, que podem ser associados a mais de um episódio. Ainda com relação aos instantes de tempo *start* e *end*, estes são definidos no domínio dos números naturais, ou seja,

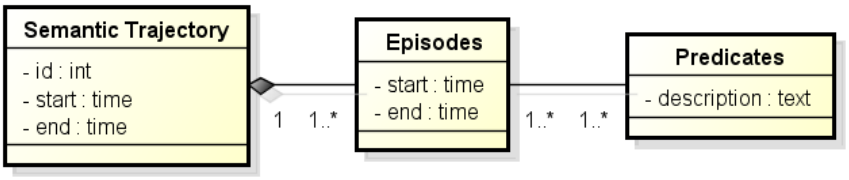


Figura 2.4: Trajetória semântica representada através de um diagrama UML.

a representação de hora terá que ser traduzida para uma representação de um único número inteiro não negativo. Resumindo, o modelo em diagrama UML descreve o conceito de trajetória semântica exposto na Definição 2.2.1. A partir desta seção, toda vez que for mencionada a palavra **trajetória**, a mesma irá se referir a uma **trajetória semântica**.

A definição de trajetórias semânticas como uma sequência de episódios, permite que uma trajetória possa desenvolver diversas sequências distintas de predicados que são válidos ao longo do tempo. Esta sequência de predicados pode ser interpretada como um **padrão de movimento** (e.g. Estava em casa, depois foi para o trabalho). Do ponto de vista formal, um padrão de movimento MP é descrito como segue.

Definição 2.2.2 (Padrão de Movimento - MP**)** *Um* ***padrão de movimento*** *é uma sequência de anotações* $MP = \{\Psi_1, \Psi_2, \cdots \Psi_k\}$ *representando que* Ψ_i *ocorre antes de* Ψ_{i+1} *em uma TS, com* $1 \leq i \leq k$*. Assim, se* Ψ_i *vale no intervalo* $[start_i, end_i]$ *e* Ψ_{i+1} *vale no intervalo* $[start_{i+1}, end_{i+1}]$*, então* $end_i \leq start_{i+1}$*.*

A partir das definições expostas acima, verificou-se o problema de como realizar consultas, que expressem um padrão de movimento, objetivando recuperar trajetórias semânticas que correspondam a este padrão. Além disso, restrições temporais poderiam flexibilizar a busca por trajetórias que contenham padrões mais específicos. Como pode ser observado na Figura 2.5, este problema terá como entradas um conjunto de trajetórias semânticas e uma consulta que irá expressar um padrão de movimento. Ao final, todas as consultas que possuírem padrões de movimento que correspondam à consulta de entrada, devem ser retornadas.

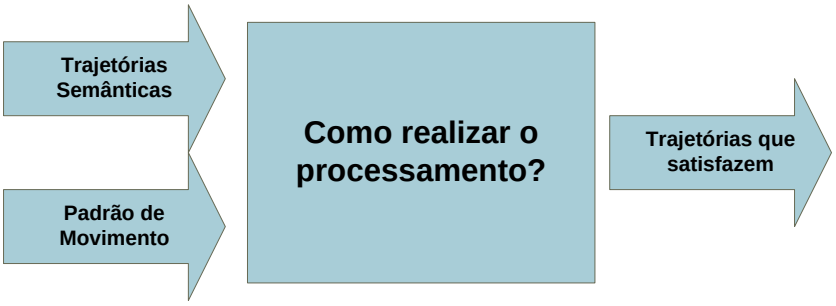


Figura 2.5: Visão geral do problema

A solução do problema apresentado por este trabalho será alcançada após diversos subproblemas serem solucionados. Na Figura 2.5, observa-se que, inicialmente, uma consulta deve ser expressa em alguma linguagem e que trajetórias semânticas devem estar armazenadas

em um banco de dados. Em relação à consulta, esta será expressa de acordo com uma linguagem, causando, assim, a necessidade pela especificação da linguagem e sua formalização. Para as trajetórias semânticas armazenadas em banco, será necessária a definição de um modelo de dados que servirá de base para a definição da semântica da linguagem de consulta. Ao final, será necessário um procedimento que realize o processamento de consultas expressas na linguagem de consulta especificada (juntamente com sua semântica), visando à recuperação de trajetórias semânticas, descritas pelo modelo de dados, que correspondam ao padrão de movimento descrito na consulta.

Com a finalidade de facilitar a compreensão do problema, considere as trajetórias semânticas, descritas como uma sequência de episódios, apresentadas na Figura 2.6, onde em cada intervalo de tempo possui uma ou mais anotações semânticas.

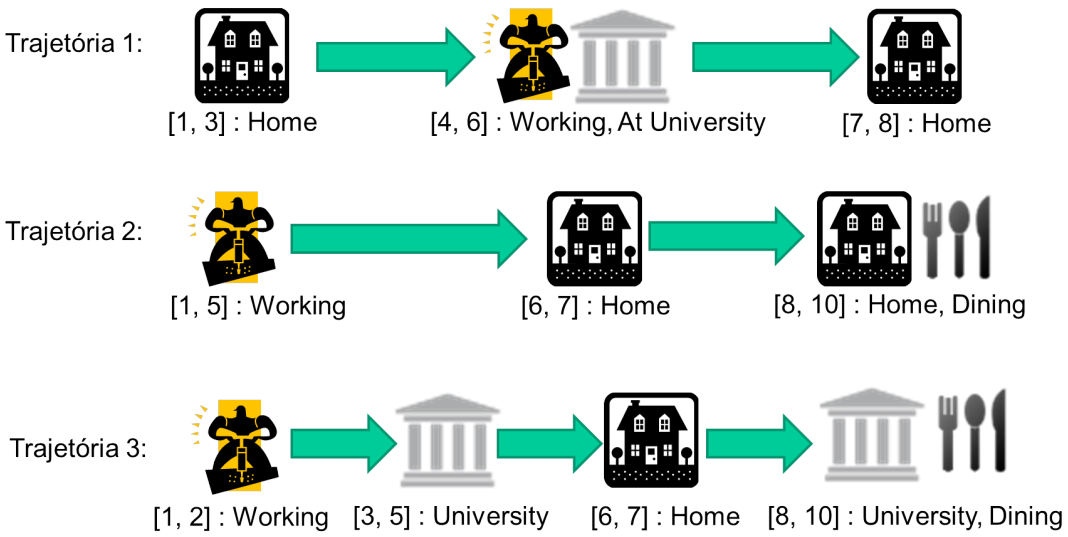


Figura 2.6: Exemplo de Trajetórias Semânticas

A partir de trajetórias semânticas representadas como na Figura 2.6, algumas consultas interessantes podem ser realizadas, como a que segue:

> *Q_1: Quais as trajetórias possuem o seguinte padrão de movimento: partem de casa, depois vão para a universidade e, então, retornam para casa?*

Para esta consulta, a única trajetória onde esta propriedade é verdadeira é a de identificador 1, que contém o padrão de movimento descrito na consulta. A Trajetória 1 permanece em casa entre as unidades de tempo 1 e 3, então se dirige a universidade, permanecendo lá entre as unidades de tempo 3 e 5. Ao final, a Trajetória 1 finaliza seu trajeto em casa, onde permanece entre as unidades de tempo 5 e 6. As outras trajetórias não satisfazem, por não conterem o padrão de movimento especificado na consulta.

Outro tipo de consulta poderia incluir restrições temporais, como restrições de duração de tempo da ocorrência de uma determinada anotação semântica. Um exemplo deste tipo

de restrição foi citado no Capítulo 1: “realizam paradas de no máximo 3 dias”. A consulta apresentada abaixo inclui estas restrições temporais.

> *Q_2: Quais trajetórias que possuem o seguinte padrão de movimento: trabalham durante pelo menos 5 unidades de tempo, depois jantam e, logo após, estão em casa?*

Para esta consulta, a única trajetória que irá satisfazer o padrão de movimento é a de identificador 2. A Trajetória 2 possui uma anotação de Trabalho (*Working*) entre os instantes de tempo 1 e 5, atendendo a restrição “durante pelo menos 5 unidades de tempo”. É possível notar que esta trajetória possui o intervalo de tempo $[5,7]$ no qual tanto a anotação referente a estar em casa (*Home*), como a anotação referente a jantar (*Dining*) são válidas. Desta forma, podemos inferir que a atividade “jantar” ocorre antes de “estar em casa”, pois, por exemplo, a ocorrência de *Dining* pode ser verificada no intervalo $[5,6)$ e a ocorrência de $Home$ pode ser verificada no intervalo $[9,10]$, apesar de as duas anotações serem válidas durante todo o intervalo $[6,7]$.

Por fim, outro tipo de consulta poderia incluir restrições temporais também entre as ocorrências de duas anotações, indicando o tempo transcorrido desde a verificação de uma anotação semântica até a próxima. Para ficar mais claro, considere a consulta abaixo que inclui esta nova restrição.

> *Q_3: Quais trajetórias que possuem o seguinte padrão de movimento: realizam uma atividade de trabalho durante pelo menos 2 unidades de tempo, então, depois de no máximo 4 unidades de tempo, estão em casa e, após isto, jantam?*

Para esta última consulta tanto a trajetória 2 quanto a 3 satisfazem o padrão de movimento descrito. Pode-se notar que, para as duas trajetórias, a ocorrência da anotação “estar em casa” (*Home*) acontece após no máximo 4 unidades de tempo. Para o caso da trajetória 2, ela acontece logo após uma atividade de trabalho, já no caso da trajetória 3, $Home$ ocorre após 4 unidades de tempo, estando de acordo com a restrição.

2.3 Conclusão

Após a explanação anterior, verificou-se que para buscar trajetórias semânticas (como a apresentada na Seção 2.2) que correspondam a um padrão de movimento descrito por uma consulta, diversos subproblemas devem ser solucionados. Abaixo, cada um dos subproblemas encontrados foi descritos como uma pergunta. Assim, esta dissertação tem como objetivo responder a todos estes questionamentos, sanando assim, senão todos, a maior parte dos subproblemas. Portanto, os subproblemas são:

1. Qual o modelo de dados utilizado para armazenar trajetórias semânticas?
2. Considerando que exista o modelo de dados do questionamento 1, qual a linguagem de consultas, que descreve padrões de movimento, a ser utilizada sobre este modelo?

3. Considerando que existe a linguagem do questionamento 2, qual a sua semântica?
4. Considerando que os questionamentos 1, 2 e 3 foram respondidos, como realizar o processamento da linguagem?
5. Considerando que todos os questionamentos anteriores foram sanados, até que ponto a solução é deficiente?

3 TRABALHOS RELACIONADOS

3.1 Introdução

A gestão de dados de trajetórias tem sido bastante estudada por diversas comunidades de pesquisa. Em banco de dados, o primeiro trabalho a propor uma implementação completa de um banco de dados para lidar com dados de objetos móveis foi proposto por (GÜTING et al., 2000). Neste trabalho, o foco principal não era tratar de trajetórias, mas de tipos de dados que podem variar ao longo do tempo, onde a trajetória é vista como uma função sobre um tipo de dado móvel. Posteriormente, (PELEKIS et al., 2006) construiu um *framework* para representar e consultar dados de objetos móveis sobre o SGBD Oracle. Técnicas para análise e gerenciamento de objetos móveis têm sido usadas pela comunidade de banco de dados para dados que consideram aspectos geométricos e temporais. Entretanto, análises que envolvem informações semânticas e restrições temporais não têm sido realizadas a contento.

Apesar dos inúmeros trabalhos na área de trajetórias de objetos móveis, neste capítulo, abordaremos os trabalhos relacionados com a modelagem de trajetórias semânticas, métodos para correspondência de padrões de movimentação, e linguagens de consulta sobre trajetórias de objetos móveis. Esses assuntos estão diretamente relacionados com o desenvolvimento de uma linguagem de consulta para padrões de movimentação sobre trajetórias semânticas.

Este capítulo está estruturado da seguinte forma. Na seção 3.2 apresentamos os trabalhos relacionados com a modelagem de trajetórias semânticas. Em seguida, na seção 3.3, destacamos os trabalhos sobre correspondências entre padrões em banco de dados. Na seção 3.4, detalhamos os principais trabalhos relacionados com linguagens de consultas para dados de trajetórias. Finalmente, na seção 3.5 concluímos este capítulo.

3.2 Modelo de Dados para Trajetórias Semânticas

Alguns trabalhos enfatizam a necessidade de abordar a análise de trajetórias semânticas (SPACCAPIETRA et al., 2008; ALVARES et al., 2007b; CAO; CONG; JENSEN, 2010). Estes trabalhos enfatizam, principalmente, a construção de modelos semânticos e a descoberta de conhecimento em trajetórias. No trabalho (SPACCAPIETRA et al., 2008) é proposto um modelo semântico para trajetórias, baseado em episódios de *stops* e de *moves*, no qual as paradas representam pontos espaço-temporais, onde ocorreu pouca ou nenhuma movimentação ao longo do tempo, e os movimentos representam o período no qual o objeto móvel estava em movimento. Em (ALVARES et al., 2007b), objetiva-se a inserção de informações semânticas às trajetórias com base no contexto geográfico da aplicação, dividindo a trajetória em regiões de paradas e trechos de movimento. Neste contexto, alguns trabalhos visam à incorporação de semântica sobre dados de trajetórias (BAGLIONI et al., 2008; XIE; DENG; ZHOU, 2009; YAN et al., 2010, 2011).

Em (XIE; DENG; ZHOU, 2009), é proposto um método que associa atividades a um determinado trecho de uma dada trajetória, baseado na sua proximidade de um determinado

ponto de interesse (POI) e na duração que a trajetória permanece próximo ao POI. Um POI é uma localização específica que alguém julgou interessante, dependendo da aplicação na qual as trajetórias estão inseridas, tal como restaurantes, estádios, shoppings, danceterias, etc. Já nos trabalhos de Yan (YAN et al., 2010, 2011), é proposto o enriquecimento semântico de trajetórias brutas, representando estas através de uma sequência de episódios que possuem um conjunto de anotações semânticas. Uma anotação semântica indica qual atividade, meio de transporte ou localização está relacionada com um dado episódio. A partir dos *stops* e dos *moves* computados, anotações semânticas são aplicadas a cada episódio, baseadas na informação semântica do espaço geográfico utilizado na aplicação.

Outro trabalho interessante neste sentido é o proposto por (BOGORNY; KUIJPERS; ALVARES, 2009). Neste trabalho é proposta uma linguagem de consultas, a ST-DMQL (*Semantic Trajectory Data Mining Query Language*), que possui diversas funcionalidades para transformar trajetórias brutas em trajetórias semânticas de um alto nível de abstração.

Apesar de trabalhos abordarem o enriquecimento semântico de trajetórias brutas, poucos trabalhos buscam a realização de consultas sobre trajetórias semânticas. Visando solucionar este problema, foi investigado trabalhos em duas áreas: correspondência de padrões em sequências de eventos e realização de consultas em trajetórias.

3.3 Correspondência de Padrões

Visando verificar a ocorrência de determinados comportamentos em um conjunto de trajetórias, foi considerada a área referente à verificação de padrões através de consultas a dados armazenados. Desta forma, foram analisados trabalhos relacionados à correspondência de padrões sobre sequências de linhas de dados (ZEMKE et al., 2007; DINDAR, 2008) e sobre uma sequência de eventos (CADONNA; GAMPER; BÖHLEN, 2011). As correspondências de padrões dos trabalhos anteriores referem-se à ocorrência de determinadas condições sobre os atributos de uma tupla de dados.

Em (ZEMKE et al., 2007), é proposto o padrão ANSI 2007 que visa buscar padrões em sequências de tuplas de dados. A motivação surge na área de negócios, na área de aplicações de segurança, entre outras, objetivando a detecção de determinados comportamentos, que podem corresponder, por exemplo, a uma ação criminosa. Para estes tipos de averiguações, este trabalho define algumas estruturas adicionais ao padrão SQL 2006, nas quais o padrão de comportamento a ser buscado é definido usando a sintaxe de Expressões Regulares. Entretanto, este trabalho não aborda uma implementação para o padrão proposto. Este trabalho não possibilita verificar se um conjunto de predicados ocorreu em um determinado intervalo de tempo.

Uma implementação de alguns operadores do trabalho anterior, voltada a estender sistemas de banco de dados relacionais com a capacidade de verificar padrões sobre sequências de linhas armazenadas, pode ser verificada em (DINDAR, 2008). Este trabalho tem como motivação a correspondência de padrões a dados armazenados em banco de dados relacionais. Os padrões são definidos através de expressões regulares que irão verificar se um determinado padrão sequencial ocorre em um banco de dados. Em decorrência de o padrão SQL não pro-

porcionar correlações entre as linhas de uma tabela nativamente, este trabalho apresenta uma extensão ao SQL usando uma máquina de estados finitos, implementada no MySQL, que possibilita a correspondência de padrões sobre uma sequência contínua de tuplas em uma tabela do banco de dados relacional. O processamento da consulta é baseado no conceito de janelas deslizantes, que, de forma simples, consiste avançar para uma linha posterior quando não se consegue encontrar todo o padrão especificado a partir da linha anterior. No contexto de padrões de movimento, este trabalho possibilitaria verificar a ocorrência de padrões sequenciais em uma tabela, possibilitando restrições temporais, embora não possibilitasse selecionar as trajetórias que conteriam este padrão. Outro limitante do trabalho refere-se a não ser possível verificar se um conjunto de predicados ocorre em um mesmo intervalo de tempo, como prevê o padrão de movimento definido no Capítulo 2. Resultados sobre o desempenho computacional do trabalho citado não foram verificados.

Em (CADONNA; GAMPER; BÖHLEN, 2011) é proposta uma abordagem para a verificação de correspondência de padrões em conjunto de eventos sequenciados. Correspondência de padrões em eventos consiste em uma técnica de consulta onde uma sequência de eventos de entrada é comparada a um padrão complexo que especifica restrições na ordem dos eventos, seus valores e quantificadores. Para representar um padrão o trabalho utiliza uma linguagem baseada em expressões regulares que será processada sobre relações que contenha um atributo referente ao instante de tempo que um evento ocorreu. Desta forma, esta abordagem não lida com intervalos de tempo, não possibilitando a verificação de padrões em sequência de intervalos. Para o processamento da consulta, é proposto um algoritmo de avaliação baseado em autômato. Este algoritmo avalia se uma determinada sequência de eventos corresponde a um determinado padrão que será traduzido para um autômato. Pode-se enfatizar que, nesta abordagem, as restrições de tempo não são um fator crucial e que não é permitido verificar se mais de um predicado é válido durante um mesmo intervalo. Outro ponto importante a ressaltar é que o algoritmo da abordagem realiza a avaliação da consulta em tempo exponencial.

3.4 Linguagens de Consultas para Trajetórias

Nesta seção serão abordados trabalhos que visam à realização de consultas sobre trajetórias que envolvam informação semântica.

No trabalho de (VIEIRA; BAKALOV; TSOTRAS, 2010), a correspondência de padrões ocorre através de uma linguagem, baseada em expressões regulares, definida através de uma gramática. As trajetórias utilizadas neste trabalho são semelhantes às trajetórias brutas definidas no início da Seção 2.1, ou seja, são carentes de informações semânticas. Para a realização de consultas são considerados somente predicados espaciais, definidos pelos relacionamentos topológicos propostos em (GÜTING; SCHNEIDER, 2005), que acontecem durante um determinado intervalo de tempo. Este intervalo de tempo pode ou não ser fornecido na construção da consulta. A linguagem de consulta proposta é baseada em expressões regulares, definida através de uma simples gramática, que pode expressar uma sequência de predicados usando um alfabeto que representa as regiões do espaço geográfico utilizado na aplicação. Este espaço geográfico (e.g. uma cidade) é dividido em regiões que não se sobrepõe, possibilitando

que a trajetória seja visualizada como uma sequência de regiões visitadas. A avaliação da consulta, de forma a recuperar as trajetórias que correspondam a um determinado padrão, pode ser realizada através de duas abordagens baseadas em estruturas de indexação. Uma grande vantagem do trabalho de (VIEIRA; BAKALOV; TSOTRAS, 2010) é que as abordagens proposta realizam a avaliação em tempo polinomial, no pior caso. Entretanto, este trabalho não lida com trajetórias semânticas como objetos de primeira classe e não permite expressar restrições de duração como "trabalhando por no máximo 5 horas" ou "trabalhando por no mínimo 3 horas". Outra desvantagem está relacionada a não possibilidade de verificar a ocorrência de mais de um predicado por intervalo de tempo, já que se tem somente uma região visitada durante um intervalo de tempo.

Ainda relacionado com a realização de consultas sobre trajetórias, em (GOMEZ; VAISMAN, 2009) é apresentada uma linguagem baseada em expressões regulares com o objetivo de refinar padrões encontrados durante um processo de mineração de dados. A representação das trajetórias utilizadas é semelhante ao conceito de trajetória bruta mencionado no início da Seção 2.1, entretanto, a avaliação da consulta ocorre sobre uma representação semântica de cada trajetória. Esta representação é obtida através do processamento dos pontos de parada (i.e. *stops*). A linguagem de consultas, que é definida através de uma gramática simples, permite verificar se determinados valores de atributos acontecem ao longo de pontos de interesse visitados por uma trajetória. Para realizar a avaliação de consultas é construído um autômato de estados finitos para avaliar quais sequências de pontos de interesses visitados por uma trajetória correspondem a um padrão expresso na linguagem. Esta avaliação de consultas é realizada em tempo exponencial. Outro aspecto relevante deste trabalho é que a avaliação de consultas proposta permite restrições temporais nos valores de atributos referentes a tempo, apesar de não possibilitar restrições de duração de tempo de um determinado predicado.

Outro trabalho relacionado às consultas sobre trajetórias é apresentado em (SAKR; GÜTING, 2010), no qual é proposta uma nova abordagem para expressar e avaliar padrões de consultas espaço-temporais em banco de dados de objetos móveis. O trabalho aborda a avaliação de consultas de forma integrada com SQL e com um otimizador de consultas através da implementação da abordagem proposta na plataforma SECONDO[1]. As trajetórias consideradas por este trabalho não são trajetórias semânticas de primeira classe, como as abordadas em (SPACCAPIETRA et al., 2008), elas são definidas através de uma função (*moving point*), tal que dado um intervalo de tempo, obtêm-se a representação geográfica da trajetória. Os padrões definidos por este trabalho especificam restrições de ordem temporal e outros tipos de restrições temporais entre predicados, como duração de tempo. A avaliação dos padrões de consultas espaço-temporais é vista como uma instância do problema de satisfação de restrições, cuja solução apresentada realiza a avaliação em tempo e espaço exponencial. Entretanto, visando um bom desempenho em grandes bancos de dados, é proposta a utilização de índices, assim como o uso de um otimizador de consultas. Assim, os experimentos realizados demonstraram que a abordagem pode realizar a avaliação de consultas de forma linear, apesar de que foram utilizados poucos predicados (no máximo 8) dependentes do tempo nos padrões de consultas

[1] Um banco de dados extensível que suporta, especialmente, aplicações não padrões, como banco de dados de objetos móveis. Mais em: `http://dna.fernuni-hagen.de/secondo/index.html`

espaço-temporais gerados para a realização das consultas. Um ponto positivo da abordagem de (SAKR; GÜTING, 2010) é a permissão da verificação de sobreposição entre predicados, assim como a possibilidade de analisar se restrições temporais, em predicados e entre eles, ocorrem.

3.5 Conclusão

Apresentamos neste capítulo os principais trabalhos relacionados com o tema desta dissertação. Apesar das várias estratégias para processamento de consultas sobre dados de trajetórias apresentadas na literatura, nenhuma das abordagens propõe uma solução para consultas sobre trajetórias semanticamente anotadas, a qual atenda os requisitos levantados no Capítulo 2. Como forma de apresentar as principais características de cada abordagem, na Tabela 3.1 é exposto um quadro comparativo entre as abordagens publicadas e discutidas anteriormente. Desta forma, o nosso trabalho visa propor uma solução que visa atender a necessidade de um processamento de consultas eficiente para uma linguagem de consulta expressiva que possibilita escrever padrões de movimento sobre trajetórias semânticas. Além disso, a solução proposta deverá atender a todos os requisitos levantados no Capítulo 2, preenchendo a lacuna deixada pelos trabalhos publicados.

Tabela 3.1: Analise comparativa entre os trabalhos relacionados.

Características	**Dindar**	**Cadonna**	**Vieira**	**Guting**	**Vaisman**
Predicados com restrições temporais	Sim	Não	Restrição de intervalo onde ocorreu determinado predicado.	Restrição de duração e de instante de tempo em predicados e entre eles	Restrições temporais nos valores de atributos de tempo
Solução voltada para Trajetórias	Não	Não	Trajetórias Brutas	Moving Point (função que dado um intervalo de tempo, obtêm-se a representação geométrica da trajetória)	Brutas, com avaliação sobre uma representação semântica
Tipo de processamento de consultas	Baseada em autômato	Baseada em autômato	Baseada em estrutura de índices	Baseado na solução do problema da satisfação de restrições	Baseada em autômato
Complexidade do processamento	Não se sabe	Exponencial	Polinomial	Exponencial: sem usar índices; Polinomial: com índices (prova feita através de experimentos)	Exponencial
A consulta permite a verificação de mais de um predicado por intervalo de tempo	Não	Não	Não	Sim	Sim
Principal desvantagem	Verificar a ocorrência de mais de um predicado por intervalo de tempo	Processamento de consultas	Não lida com restrições de duração	Experimentos com poucos predicados dependentes do tempo	Processamento de consultas

4 PROCESSAMENTO DE CONSULTAS PARA PADRÕES DE MOVIMENTO

A partir das dificuldades apontadas pelos trabalhos relacionados e da busca por uma forma de realizar consultas em trajetórias semânticas, um método foi proposto. Para a definição formal de uma linguagem de consultas, foi investigado o campo da Lógica Temporal. Este campo foi escolhido devido aos padrões de movimentos, a serem verificados nas trajetórias, conterem restrições temporais, como a sequência da ocorrência de predicados, a duração de predicados e a duração entre predicados, conforme apresentado no Capítulo 2. Dentre as lógicas temporais estudadas, foi analisado o problema da Verificação de Modelos, que consiste em verificar se um determinado sistema satisfaz uma dada propriedade. A verificação de modelos se refere a um método formal utilizado na especificação, desenvolvimento e verificação de softwares e hardwares. Este método consiste em verificar se um dado sistema satisfaz uma determinada propriedade. Assim, todos os conceitos inerentes à lógica temporal e à verificação de modelos são explicados na Seção 4.1.

A partir do estudo realizado em Lógica Temporal e em Verificação de Modelos, foi averiguado que a verificação de modelos poderia ser realizada como um processo de avaliação se uma determinada trajetória satisfaz uma dada consulta (Seção 4.2). Dentro deste contexto, uma trajetória pode ser expressa por um autômato temporizado e a consulta por uma lógica temporal que possibilite a verificação de determinadas restrições, como restrições de duração de tempo. Após diversos estudos em Lógica, visando encontrar uma que expressasse as restrições temporais requeridas, optou-se por utilizar a Timed-CTL (TCTL), devido a esta possuir um subscrito com restrições de duração de tempo em suas modalidades, permitindo atender as especificações de consultas para trajetórias semânticas.

Com consultas expressas em TCTL, verificou-se que as sintaxes destas não seriam de fácil entendimento por usuários de banco de dados, já que, para o entendimento das consultas, seriam necessários conhecimentos acerca de lógicas modais, temporais e acerca da sintaxe e da semântica da lógica específica, TCTL. Para contornar esta limitação, foi proposta uma linguagem de consulta (Seção 4.3) que servirá como um *syntactic sugar* para fórmulas TCTL, ou seja, a linguagem proposta será apenas uma forma mais natural para o usuário expressar as consultas, mas a semântica real será definida pela TCTL.

Antes de descrever a solução proposta (Seção 4.2) para a realização de consultas em trajetórias semânticas e uma linguagem de consultas visando simplificar fórmulas TCTL, conceitos preliminares, necessários à compreensão da abordagem, são propostos na Seção 4.1.

4.1 Preliminares

Nesta seção são definidos os conceitos preliminares necessários à compreensão da abordagem proposta, que visa o processamento de consultas sobre trajetórias semânticas. A solução proposta na Seção 4.2 surgiu após a necessidade de se definir formalmente o processamento de consultas sobre trajetórias semânticas, visam obter aquelas trajetórias que satisfazem um determinado padrão de movimento. Como um padrão de movimento expressa uma

sequência da ocorrência de predicados, seguindo uma ordem temporal, foi analisado o campo da Lógica Temporal. Dentre as lógicas estudadas, optou-se por utilizar a TCTL, que permite a inclusão de restrições temporais. A seguir, os conceitos básicos inerentes à solução são apresentados.

4.1.1 Lógica Temporal

Lógicas temporais são de grande importância para representação de propriedades cujos valores mudam ao longo do tempo, possibilitando expressar ordem e relações casuais entre os estados das propriedades. Nas lógicas clássicas o valor verdade de uma propriedade não muda ao longo do tempo (HUTH; RYAN, 2004). Dentre as lógicas clássicas mais comuns, podemos citar a Lógica Proposicional e a Lógica de Primeira Ordem, bastante utilizadas diariamente.

Na Lógica Proposicional as fórmulas são construídas baseadas em um conjunto de fatos elementares (i.e. fórmulas atômicas), usando um conjunto de operadores lógicos $(\neg, \wedge, \vee, \rightarrow, \leftrightarrow)$. A semântica destes operadores pode ser definida através de tabelas verdade ou por regras indutivas na estrutura da fórmula, onde esta pode assumir o valor lógico verdadeiro ($\top$) ou falso ($\bot$).

A Lógica de Primeira Ordem adiciona diversas extensões à Lógica Proposicional, como um domínio de elementos onde as fórmulas lógicas são construídas, um conjunto de relações definidas sobre o domínio de elementos, predicados n-ários associados com cada uma das relações, os quais são funções que retornam valores verdadeiros ($\top$) ou falsos ($\bot$), e quantificadores existencial ($\exists$) e universal ($\forall$) (EBBINGHAUS; FLUM; THOMAS, 1994; BELLINI; MATTOLINI; NESI, 2000).

De forma geral, uma assertiva em lógica pode ser classificada como estática ou dinâmica. Assertivas estáticas têm sempre os mesmo valores, ou seja, são independentes do tempo, enquanto assertivas dinâmicas podem ter os seus valores verdade alterados ao longo do tempo, ou seja, são dependentes do tempo. Como exemplos de assertivas estáticas, tem-se: $2 < 3$, $2 = 2$, $par(2)$, etc. Exemplos de assertivas dinâmicas: "Está chovendo", "Amanhã fará sol", "Estou com sede", etc. Dentro desta classificação, as lógicas temporais são classificadas como dinâmicas, ou seja, com elas é possível expressar assertivas indicando momentos onde estas serão falsas ou verdadeiras, nas quais o tempo será modelado como uma variável explícita (BELLINI; MATTOLINI; NESI, 2000).

Lógicas temporais são um tipo específico de lógica modal (GARSON, 2009), nas quais o modelo lógico é definido pela tupla $\langle W, R, V \rangle$, tal que W é interpretado como o conjunto de todos os instantes de tempo possíveis de um domínio temporal T, $R \subseteq W \times W$ é uma função de acessibilidade entre os instantes de tempo e V é uma função de avaliação das fórmulas para um dado instante de tempo, definida como $V : F \times W \rightarrow \{\top, \bot\}$, tal que F representa o conjunto das fórmulas da lógica temporal. Esta função indicará quais as fórmulas lógicas válidas para cada instante pertencente à W. As lógicas temporais são vistas como extensões da lógica clássica, possibilitando a uma fórmula ser avaliada dinamicamente. Para permitir

a análise de que uma determinada fórmula modificará o seu valor ao longo do tempo, novos operadores foram adicionados às lógicas clássicas (BELLINI; MATTOLINI; NESI, 2000):

- **G**: sempre verdade no futuro;
- **F**: eventualmente verdade no futuro;
- **H**: sempre verdade no passado;
- **P**: eventualmente verdade no passado.

Com relação à classificação, as lógicas temporais podem ser divididas baseadas em diversos critérios, como "lógicas proposicionais ou lógicas de primeira ordem", "lógicas baseadas em ponto ou lógicas baseadas em intervalo", "lógicas de tempo linear ou lógicas de tempo ramificado" (EMERSON, 1990). Para realizar a formalização de padrões de movimento, foram estudadas lógicas temporais baseadas em ponto e baseadas em intervalo. Após diversas análises, verificou-se que, além de formalizar padrões de movimento através da Lógica Temporal, poderia ser proposto um método para o processamento de consultas que expressam estes padrões. Foi averiguado, então, que o problema de processamento de consultas com restrições temporais poderia ser interpretado como um problema de *verificação de modelos*, no qual cada trajetória representaria um modelo e a consulta representaria uma propriedade (um padrão de movimento), escrita em uma determinada lógica, que seria verificada no mencionado modelo. Como padrões de movimento necessitam que restrições temporais possam ser expressas, um estudo sobre lógicas temporais que possibilitem restrições temporais e sobre a viabilidade computacional de seu respectivo verificador de modelo, foi desenvolvido.

Após ser verificado que a lógica Timed-CTL (TCTL) possibilita restrições temporais e que a sua verificação de modelos pode ser realizada, para o caso no qual o modelo é representado por um *autômato temporizado*, em tempo polinomial (LAROUSSINIE; MARKEY; SCHNOEBELEN, 2004), optou-se por utilizá-la. Desta forma, espera-se que o processamento de consultas, através da verificação de modelos envolvendo a TCTL seja realizado em tempo polinomial, o que pode ser um diferencial em comparação com trabalhos semelhantes, como os apresentados no Capítulo 3.

A seguir, os conceitos inerentes à *verificação de modelos*, *autômato temporizado* e *TCTL*, necessários para a compreensão da solução, são fornecidos.

4.1.2 Verificação de Modelos e sua Aplicabilidade em Ciências da Computação

Verificação de Modelos se refere ao problema de realizar testes automáticos para saber se um modelo, que representa um sistema, atende uma determinada especificação. A fim de resolver este problema com relação ao aspecto computacional, tanto o modelo quanto a especificação são formuladas em uma linguagem matemática precisa. Desta forma, alguns trabalhos consideram Verificação de Modelos como um método formal de verificação para sistemas baseados em estados, que permite analisar se um sistema atende uma determinada especificação.

Este método tem sido aplicado em vários campos, como em engenharia de processo, projetos de hardware, projetos de software e verificação de protocolos (MEHLER, 2005). Os métodos formais têm sido fortemente recomendados como técnicas de verificação no desenvolvimento de sistemas críticos de segurança, sendo inclusive recomendados como "melhores práticas" pela IEC (*International Electrotechnical Commission*) (BAIER; KATOEN et al., 2008).

Na prática, um protocolo, por exemplo, pode controlar o comportamento de um determinado sistema, restringindo quais as transições de estados são permitidas nele. Assim, um protocolo poderá ser modelado em um modelo matemático, visando à realização da Verificação de Modelos para certificar se o protocolo está de acordo com sua especificação. A Figura 4.1 apresenta um exemplo, no qual um protocolo especifica o comportamento de um elevador. Para este caso, necessita-se verificar se a propriedade a seguir é verdadeira: sempre estará seguro para uma pessoa entrar, quando o botão for apertado e a porta se abrir. Desta forma, um verificador de modelos poderá verificar, para o modelo lógico, representante do protocolo, e para a fórmula lógica, representante da propriedade, se o modelo satisfaz a fórmula, ou seja, se o protocolo satisfaz a propriedade.

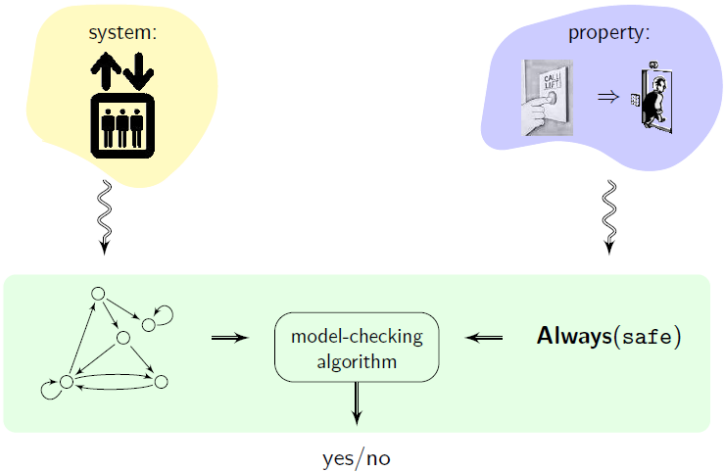


Figura 4.1: Visão geral de um Verificador de Modelos.

Do ponto de vista formal, Verificação de Modelos é uma técnica de verificação que explora todos os possíveis estados de um sistema através da "força-bruta". Esta técnica é formalizada através da Lógica Temporal, na qual o modelo será representado por um modelo lógico e a propriedade será representada por uma fórmula. A ideia de se utilizar Lógica Temporal surge do fato de que uma fórmula pode ter seu valor verdade alterado ao longo do tempo, devido ao caráter dinâmico da lógica, ou seja, as fórmulas podem mudar seus valores conforme o sistema evolui. Formalmente, dado um modelo $\mathscr{M}$ e uma propriedade ϕ, a verificação de modelos irá analisar, para cada estado s do modelo, se a propriedade ϕ vale em um destes estados. Para verificar se um modelo satisfaz uma propriedade, devem-se seguir os passos adiantes:

- modelar o sistema utilizando uma formalização lógica, construindo, assim, um modelo $\mathscr{M}$;
- codificar a propriedade a ser verificada, utilizando a linguagem de especificação do verificador de modelos, resultando em uma fórmula da lógica temporal ϕ;

- executar o verificador de modelos para as entradas $\mathscr{M}$ e ϕ.

Em Lógica Temporal, o modelo pode ser formalizado por um sistema de transição $\mathscr{T}$ baseado em Autômato Temporizado e a propriedade por uma fórmula da lógica TCTL, representado por ϕ. Assim, verificar se um modelo $\mathscr{M}$ satisfaz uma propriedade ϕ, irá se resumir a verificar se um estado do sistema de transição $\mathscr{T}$ satisfaz uma fórmula TCTL ϕ. Na Figura 4.1 o elevador poderia ser representado como o sistema de transição $\mathscr{T}$ e a propriedade como uma fórmula da lógica TCTL, *EG safe*.

A seguir, definições relacionadas a *autômato temporizado* e a lógica TCTL são apresentadas.

4.1.3 Autômato Temporizado

Um Autômato Temporizado representa um modelo matemático que torna possível expressar restrições de tempo-real através de um autômato. Este modelo foi proposto por Alur e Dill (ALUR; DILL, 1990, 1994), e consiste em estender o conceito de autômatos clássicos com um conjunto de variáveis reais - chamadas de *clock* - que aumentam sincronicamente com o tempo, associando restrições sobre estas variáveis (BOUYER; LAROUSSINIE, 2010).

Formalmente, um autômato temporizado é, essencialmente, um autômato de estados finitos (ou seja, um grafo contendo um conjunto finito de nós e um conjunto finito de arestas rotuladas) com a possibilidade de manipular *clocks*. Os *clocks* são variáveis reais que têm seus valores inicializados com zero e, quando o sistema é iniciado, estes valores são alterados sincronicamente e continuamente com o decorrer do tempo, a uma mesma taxa (BENGTSSON; YI, 2004). Além disso, o autômato temporizado possui restrições sobre as variáveis de *clock*. Estas restrições são classificadas como **restrições de guardas**, que são associadas a uma transição e especificam quando estas podem ser realizadas, de acordo com os valores de *clocks*; e como **restrições invariantes**, que são associadas a um dado estado e indicam os valores possíveis dos *clocks* para este estado. Além destas restrições, um autômato temporizado pode possuir operações de atualização dos valores de *clocks*, aplicadas quando uma transição é realizada (BOUYER; LAROUSSINIE, 2010).

Do ponto de vista teórico, seja $\mathbb{N}$ e $\mathbb{R}$ os conjuntos dos números naturais e reais, respectivamente. Assuma que $\mathscr{C}$ é o conjunto das variáveis de *clock* (e.g. x, y,...), então $\mathscr{B}(\mathscr{C})$ denota o conjunto das expressões booleanas sobre fórmulas atômicas da forma $x \sim c$ ou $x - y \sim c$, com $x, y \in \mathscr{C}$, $c \in \mathbb{N}$ e $\sim \in \{<, \leq, =, \geq, >\}$. Restrições pertencentes a $\mathscr{B}(\mathscr{C})$ são interpretadas a partir das valorações para as variáveis de clock $\mathscr{C}$ e são chamadas de **restrições de guarda** (também utilizadas como restrições invariantes) (LAROUSSINIE; MARKEY; SCHNOEBELEN, 2004). Estas valorações são definidas como funções de $\mathscr{C}$ para $\mathbb{R}^+$. Do ponto de vista formal, um autômato temporizado é definido como segue:

Definição 4.1.1 (Autômato Temporizado) *Um autômato temporizado $\mathscr{A}$ é uma tupla $(L, l_0, \mathscr{C}, E, I, V)$, na qual L é o conjunto de estados; l_0 é o estado inicial; $\mathscr{C}$ é um conjunto de variáveis de clock; $E \subseteq L \times \mathscr{B}(\mathscr{C}) \times 2^{\mathscr{C}} \times L$ é um conjunto de arestas entre os estados com restrições de*

guarda do tipo $x \sim c \in \mathscr{B}(\mathscr{C})$ e um conjunto de variáveis de clock a serem reinicializadas, representado por um conjunto pertencente a $2^{\mathscr{C}}$ (conjunto das partes de $\mathscr{C}$); e $I : L \to \mathscr{B}(\mathscr{C})$ representa uma função que atribui invariantes aos estados. Finalmente, $V : L \to 2^{AP}$ indica as fórmulas atômicas válidas em um dado estado.

A semântica de um autômato temporizado é definida como um sistema de transição no qual uma configuração consiste do estado atual e os valores atuais das variáveis de *clock*. Assim, existem dois tipos de transições: uma transição de atraso ou uma transição de ação. Uma transição de atraso indica que o autômato pode atrasar uma quantidade de tempo, mudando apenas sua configuração e não alterando o seu estado. Já uma transição de ação indica que o autômato verificou que uma determinada transição estava habilitada (a restrição de guarda está com valor verdadeiro).

Visando a definição da semântica de um autômato temporizado, sejam u, v funções de valoração de *clock*, então $u \in g$ significa que o valor denotado por u satisfaz a restrição de guarda g. Para $d \in \mathbb{R}+$, seja $u+d$ uma atribuição de *clock* que mapeia todos $x \in \mathscr{C}$ para $u(x)+d$, e para $r \subseteq \mathscr{C}$, seja $[r \mapsto 0]u$ significando a atribuição de *clock* que mapeia todas as variáveis de *clock* em r para 0 e concorda com u para todos as outras variáveis de *clock*, excetuando as variáveis em r. Do ponto de vista formal, a semântica de um autômato temporizado é definida como segue (BENGTSSON; YI, 2004):

Definição 4.1.2 (Semântica - Autômato Temporizado) *A semântica de um autômato temporizado $\mathscr{A}$ é um sistema de transição $\mathscr{T}$, onde os estados são pares $\langle l, u \rangle$, onde l é um estado e u é uma função de valoração de clocks, e as transições são definidas pelas seguintes regras:*

- ***Transição de atraso:*** *$\langle l, u \rangle \xrightarrow{d} \langle l, u+d \rangle$ se $u \in I(l)$ e $(u+d) \in I(l)$ para um número real não-negativo $d \in \mathbb{R}^+$*
- ***Transição de ação:*** *$\langle l, u \rangle \to \langle l', u' \rangle$ se $l \xrightarrow{g,r} l'$, $u \in g$, $u' = [r \mapsto 0]u$ e $u' \in I(l')$*

De forma a mostrar no que se constitui um autômato temporizado e como é sua semântica, considere a Figura 4.2. Considerando x como a única variável de *clock*, o primeiro estado da Figura 4.2 pode ter as seguintes configurações: $(l_0, x = 0)$, $(l_0, x = 1)$, $(l_0, x = 2)$ e $(l_0, x = 3)$, ou seja, transições de atraso. A transição de ação ocorre quando $x = 3$. Para maiores detalhes, veja (BOUYER; LAROUSSINIE, 2010).

4.1.4 TCTL - Uma Extensão da Lógica Temporal Ramificada

A TCTL é uma extensão da lógica de tempo ramificado CTL (*Computation Tree Logic*) (HUTH; RYAN, 2004), na qual as modalidades temporais possuem um subscrito com restrições de duração de tempo. TCTL é classificada como uma lógica proposicional, baseada em ponto e de tempo ramificado. Em decorrência desta lógica lidar com restrições de duração de tempo, optou-se por estudá-la visando solucionar o problema de avaliação de consultas que expressam padrões de movimento, definidos no Capítulo 2. A sintaxe da lógica TCTL é definida a seguir:

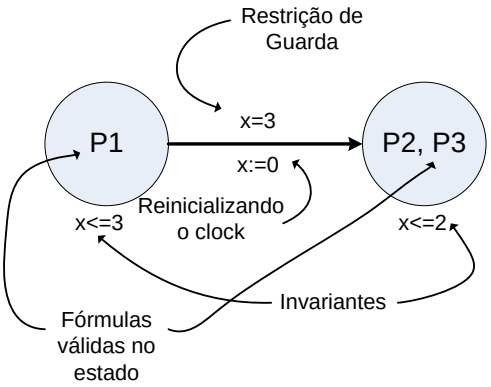


Figura 4.2: Visão geral de um Autômato Temporizado.

Definição 4.1.3 (Sintaxe da TCTL) *Uma fórmula TCTL é definida pela seguinte gramática:*

$$\varphi, \psi ::= P_1 \mid P_2 \mid \cdots \mid \neg\varphi \mid \varphi \wedge \psi \mid \exists \varphi \bigcup_{\sim c} \psi \mid A\varphi \bigcup_{\sim c} \psi$$

onde $\sim$ pode ser qualquer comparador pertencente a $\{<, \leq, =, \geq, >\}$, c é qualquer número natural e $P_i \in AP$. AP é o conjunto de proposições.

Abreviações padrões incluem $\top, \bot, \varphi \wedge \psi, \varphi \rightarrow \psi, ...$ como também as abreviações referentes aos modais, como $\mathsf{EF}_{\sim c}\varphi$ (para $\mathsf{E}\top\bigcup_{\sim c}\varphi$), $\mathsf{AF}_{\sim c}\varphi$ (para $\mathsf{A}\top\bigcup_{\sim c}\varphi$), $\mathsf{EG}_{\sim c}\varphi$ (para $\neg\mathsf{AF}_{\sim c}\neg\varphi$) e $\mathsf{AG}_{\sim c}$ (por $\neg\mathsf{EF}_{\sim c}\neg\varphi$). Ainda com relação às modalidades, se elas aparecerem sem o subscrito, $\bigcup$, F e G, significa que tem o mesmo significado que $\bigcup_{\geq 0}$, $\mathsf{F}_{\geq 0}$ e $\mathsf{G}_{\geq 0}$. O tamanho $|\varphi|$ da fórmula φ é definida de forma padrão, com as constantes escritas na notação binária.

A semântica da TCTL pode ser definida através de **Autômatos Temporizados**. Para a definição da semântica da lógica TCTL, alguns conceitos devem ser definidos. Sejam $\xrightarrow{d}$ e $\rightarrow_a$ as representações, respectivamente, de transição de atraso e de transição de ação, e seja $s = (l, u)$ uma configuração, como definidas na Seção 4.1.3. Uma execução ρ iniciada na configuração s pode ser descrita como uma sequência infinita $s = s_0 \xrightarrow{d_0}\rightarrow_a s_1 \xrightarrow{d_1}\rightarrow_a \cdots$ para algum $d_i \in \mathbb{R}^+$. A execução ρ atravessa qualquer configuração s' alcançável a partir de um caminho até s_i, por uma transição de duração $d \in [0, d_i]$. Quando isto acontece, nós escrevemos $s' \in \rho$. Seja $Exec(s)$ o conjunto de todas as execuções iniciadas em s. Assim, seja $\rho \in Exec(s)$, ou seja, uma execução pertencente a todas as execuções iniciadas em s, qualquer prefixo σ levando a uma configuração s' é descrito por $s \xmapsto{\sigma} s'$ e terá uma duração, $Time(s \xmapsto{\sigma} s')$, definida como a soma de todos os atrasos ocorridos ao longo de σ. Seja $Pref(\rho)$, todos os prefixos de ρ (LAROUSSINIE; MARKEY; SCHNOEBELEN, 2004).

Assim, seja $\rho \in Exec(s)$ e $s', s'' \in \rho$, s' precede estritamente s'' ao longo de ρ (escrito como $s' <_\rho s''$) se e somente se existe uma sub-execução σ em ρ tal que $s' \xmapsto{\sigma} s$" e σ contém pelo menos uma transição de atraso não-nula ou uma transição de ação. Pode-se notar, então, que uma mesma configuração pode ter diversas ocorrências ao longo de ρ, como por exemplo: $s <_\rho s$ ou $s <_\rho s'$ e $s' <_\rho s$ (BOUYER; LAROUSSINIE, 2010).

Definição 4.1.4 (Semântica TCTL) *As cláusulas abaixo definem quando uma configuração s,*

de um sistema de transição $\mathscr{T}$, satisfaz uma fórmula TCTL φ, escrito como $s \models \varphi$, por indução sobre a estrutura de φ (a semântica dos operadores booleanos, da lógica clássica, foi omitida):

$$s \models \mathsf{E}\varphi \bigcup_{\sim c} \psi \quad sse\ \exists \rho \in Exec(s)\ com\ \rho = \sigma.\rho'\ e\ s \xmapsto{\sigma} s',\ tal\ que\ Time(s \xmapsto{\sigma} s') \sim c,\ s' \models \psi\ e\ \forall s'' <_\rho s',\ s'' \models \varphi$$

$$s \models \mathsf{A}\varphi \bigcup_{\sim c} \psi \quad sse\ \forall \rho \in Exec(s),\ \exists \sigma \in Pref(\rho),\ tal\ que\ s \xmapsto{\sigma} s',\ Time(s \xmapsto{\sigma} s') \sim c,\ s' \models \psi\ e\ \forall s'' <_\rho s',\ s'' \models \varphi$$

De forma simples, $\mathsf{E}\varphi \bigcup_{\sim c} \psi$ indica que existe uma possibilidade de que φ irá valer até ψ valer, respeitando a restrição temporal de que a soma dos atrasos de tempo, nos quais φ vale, satisfaça a restrição $\sim c$. Enquanto $\mathsf{A}\varphi \bigcup_{\sim c} \psi$ indica que, para qualquer caso, sempre φ irá valer até ψ valer, respeitando a restrição temporal de que a soma dos atrasos de tempo, nos quais φ vale, satisfaça a restrição $\sim c$.

4.2 Avaliação de Consultas baseada em Verificação de Modelos

Nesta seção, é proposto um método de avaliação de consultas que expressam padrões de movimento. Este método é baseado em verificação de modelos em TCTL. Esta abordagem consiste em construir um sistema para cada par trajetória $\times$ consulta, no qual uma dada trajetória será modelada como um *autômato temporizado* e a consulta será expressa como uma fórmula TCTL. Para descobrir quais trajetórias satisfaz a uma dada consulta, uma verificação de modelos será realizada em cada sistema constituído de um autômato temporizado, representativo de uma trajetória, e da fórmula TCTL representativa da consulta. Ou seja, as entradas do verificador de modelos será o autômato temporizado e a fórmula TCTL. Para cada sistema cuja verificação de modelos responder positivamente, a trajetória relacionada ao autômato será incluída no conjunto resultado. Como forma de deixar claro a proposta deste trabalho, veja a Figura 4.3. Esta figura mostra que uma consulta será traduzida para uma fórmula TCTL e que cada trajetória do banco de dados será traduzida para um *autômato temporizado*. Então, as trajetórias (i.e. os autômatos) que responderem verdadeiro para a verificação de modelos, com relação à fórmula TCTL, serão retornadas.

Na Figura 4.3 um sistema (6) é uma abstração para as entradas do verificador de modelos (7). A abordagem consiste em, a partir de uma consulta Q_1 (1), que envolve padrões de movimento, representá-la em uma fórmula TCTL (2) que servirá de entrada para um sistema (6). Ao mesmo tempo, as trajetórias pertencentes a um banco de dados (3) serão traduzidas para autômatos temporizados (4). Posteriormente, para cada autômato temporizado referente a uma trajetória (5), este fará parte de um sistema (6), juntamente com a fórmula da lógica TCTL (2). Em seguida, o verificador de modelos (7) será executado para as entradas pertencentes ao sistema (6). Se o verificador de modelos retornar verdadeiro, então a trajetória semântica será incluída no resultado (9) e, então, será verificado se ainda existe alguma outra trajetória no banco de dados (10). Caso ainda existam trajetórias no banco de dados, o mesmo procedimento será efetuado para o autômato representativo da próxima trajetória e para a mesma fórmula TCTL. Caso não existam mais trajetórias no banco de dados, o processamento da consulta é

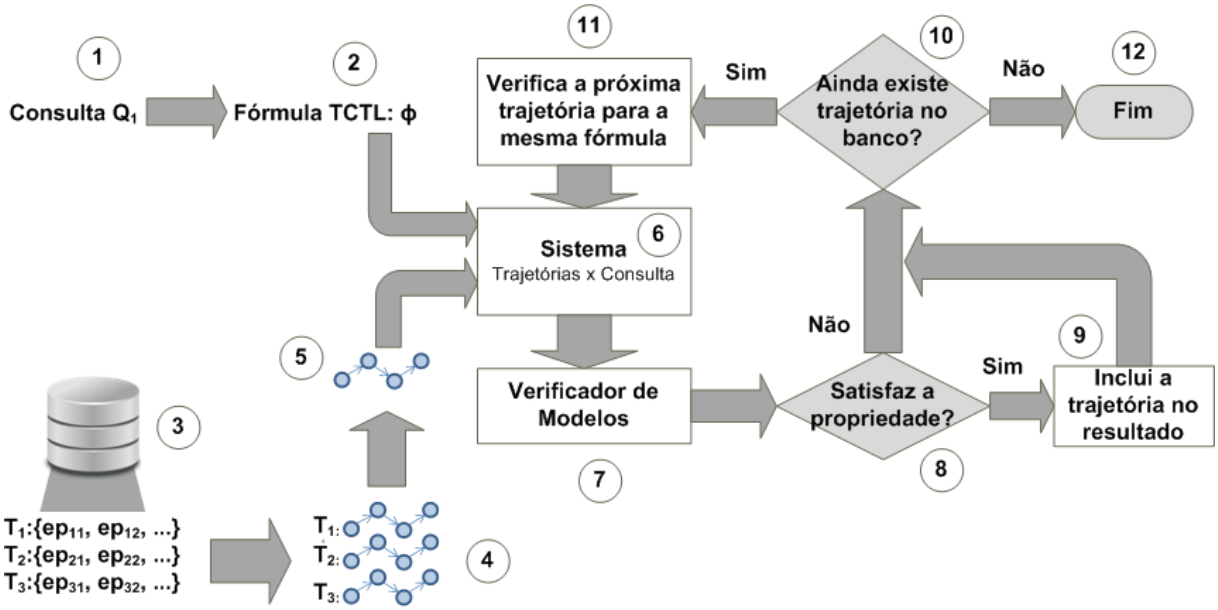


Figura 4.3: Visão geral da solução proposta.

finalizado. Na ocasião de o modelo não satisfazer a fórmula (8), a trajetória não será incluída no conjunto resultado e será verificado se ainda existe alguma outra trajetória no banco de dados (10), repetindo o processo explicado anteriormente, de forma semelhante.

A tradução de uma trajetória semântica, definida pelo modelo do Capítulo 2, para um *autômato temporizado* $\mathscr{A} = (L, l_0, \mathscr{C}, E, I, V)$ será realizada através do Algoritmo 1, que recebe como entrada uma trajetória semântica TS definida como uma sequência ordenada de episódios. Inicialmente, o algoritmo inclui x ao conjunto de variáveis de *clock*. Para cada episódio $ep \in TS$, será criado um estado l, que será inicial, caso o episódio ep seja o primeiro da trajetória. Se ep não for o primeiro episódio, então se cria uma transição do estado referente ao episódio predecessor ($l_{predecessor}$) para o estado atual (l), com uma restrição de guarda igual ao instante de tempo final do episódio predecessor (i.e. $x = predecessor(ep).end$), e inclui esta transição no conjunto de transições E do *autômato temporizado* $\mathscr{A}$. Posteriormente, o algoritmo inclui, como invariante do estado criado l, o valor $x \leq ep.end$ e faz o valor de $V(l) = ep.\Psi$, deixando todos os predicados pertencentes à ep serem válidos em l. Ao fim do laço, o Algoritmo 1 inclui o estado l no conjunto de estados do modelo $\mathscr{A}$. Após fazer este procedimento para cada ep, o algoritmo é finalizado ao retornar o *autômato temporizado* construído.

No autômato $\mathscr{A}$, cada estado $l \in L$ irá representar um episódio $ep = (start, end, \Psi)$, tal que $I(l) = x \leq end$ e $V(l) = \Psi$ (i.e. os predicados que valem para o respectivo episódio). O primeiro estado do autômato (l_0) irá se referir ao primeiro episódio da trajetória. As transições E de um autômato serão definidas de acordo com a relação de sucessão entre episódios. Considere dois episódios $ep_1 = (start_1, end_1, \Psi_1)$ e $ep_2 = (start_2, end_2, \Psi_2)$, tal que ep_2 é o sucessor de ep_1, então será construída uma transição do estado que representa o primeiro episódio (i.e. ep_1) l_{ep_1} para o estado que representa o segundo episódio (i.e. ep_2) l_{ep_2}, tal que a restrição de guarda g seja $x = end_1$. De uma maneira simples, a aresta explicada anteriormente, ficaria desta forma: $l_{ep_1} \xrightarrow{g} l_{ep_2}$.

Algoritmo 1: Construção do *autômato temporizado* da trajetória

Entrada: Uma trajetória TS definida como uma sequência ordenada de episódios $ep_i = (start_i, end_i, \Psi_i)$, tal que $1 \leq i \leq n$, cuja quantidade de episódios é n

Saída: Um *autômato temporizado* $\mathscr{A}$.

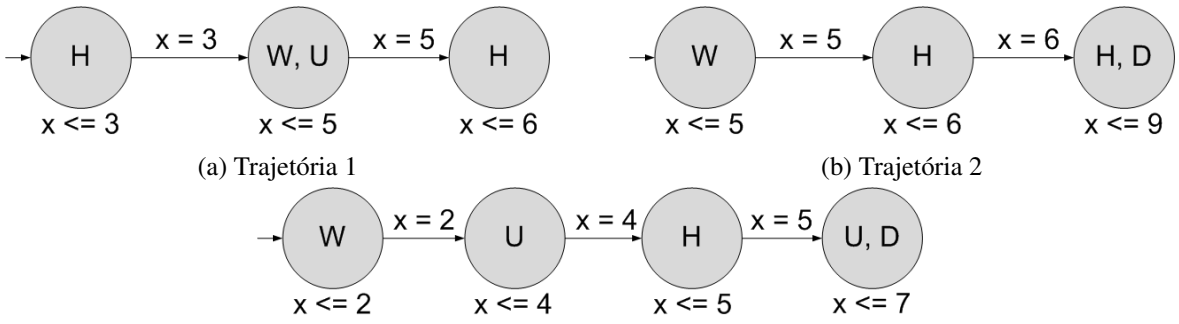
1 $\mathscr{C} \leftarrow x$; `// Inclui` x `ao conjunto de variáveis de clock`
2 **para** ep $\in TS$ **faça**
3 Crie um estado l;
4 **se** ep *é o primeiro episódio* **então**
5 $l_0 \leftarrow l$;
6 **senão**
7 $l_{predecessor} \leftarrow$ o estado referente ao `predecessor(`ep `)`;
8 Crie uma transição trans $= \mathsf{l}_{\mathsf{predecessor}} \xrightarrow{\mathsf{guarda}} \mathsf{l}$ com **guarda**="$x =$ `predecessor(`ep `).`*end*"; `// Ou seja, uma transição do estado correspondente ao episódio predecessor, para o estado correspondente ao episódio atual`
9 Inclua trans no conjunto E do *autômato temporizado* $\mathscr{A}$;
10 **fim se**
11 Inclua como invariante (I) de l o valor $x \leq$ ep.*end*;
12 Faça `V(`l`)` $=$ ep.ψ; `// Ou seja, todos os predicados que pertence ao` ep `farão parte de V(`l`)`
13 Inclua l no conjunto L do *autômato temporizado* $\mathscr{A}$;
14 **fim para**
15 **retorna** *autômato temporizado* $\mathscr{A} = (L, l_0, \mathscr{C}, E, I, V)$;

Visando apresentar a modelagem de trajetórias em *autômatos temporizados*, considere as trajetórias apresentadas no Capítulo 2, seus respectivos autômatos são apresentados na Figura 4.4. As trajetórias são apresentadas abaixo:

- **Trajetória 1:** $\{(1,3,[Home]), (3,5,[Working, At\_University]), (5,6,[Home])\}$;
- **Trajetória 2:** $\{(1,5,[Working]), (5,6,[Home]), (6,9,[Home, Dining])\}$.
- **Trajetória 3:** $\{(1,2,[Working]), (2,4,[At\_University]), (4,5,[Home]), (5,7, [At\_University, Dining])\}$;

Com relação às consultas, estas serão expressas em TCTL, já que a mesma pode expressar restrições de duração de tempo que um determinado predicado ocorreu ou duração de tempo entre a ocorrência de predicados. Por exemplo, $\mathsf{EF}(Home)$ significa que, em algum instante no futuro, a partir de um estado inicial, o predicado *Home* (estar em casa) será verdadeiro. Um exemplo no qual ocorre uma restrição temporal pode ser a fórmula $\mathsf{EF}_{\geq 2}(Home)$, a qual indica que, em algum instante no futuro, após no mínimo 2 instantes de tempo de um estado inicial, o predicado *Home* será verdadeiro.

A partir da sintaxe e da semântica da TCTL, exposta na Seção 4.1.4, foi realizada a representação das consultas citadas no Capítulo 2 para fórmulas TCTL, como apresentado abaixo:



(a) Trajetória 1

(b) Trajetória 2

(c) Trajetória 3

Figura 4.4: Trajetórias modeladas como um *autômato temporizado*

Q_1: Quais as trajetórias possuem o seguinte padrão de movimento: partem de casa, depois vão para a universidade e, então, retornam para casa?

Esta consulta pode ser expressa, em TCTL, da seguinte forma:

$$\mathsf{EF}(\mathsf{E}\, Home \bigcup_{\geq 1} \mathsf{EF}(\mathsf{E}\, At\_University \bigcup_{\geq 1} \mathsf{EF}(Home)))$$

Ou seja, a partir de um estado inicial, existirá, em algum instante de tempo posterior, uma configuração na qual o predicado *Home* é verdadeiro em pelo menos 1 unidade de tempo. O predicado *Home* será válido até existir, em um tempo posterior, o predicado *At_University* em pelo menos 1 unidade de tempo até a ocorrência de *Home* novamente. O modal EF precedendo os predicados após o modal $\bigcup$ indica que a sucessão de predicados ao longo do tempo, em uma trajetória, não precisa, necessariamente, indicar que um ocorre imediatamente ao outro.

Considere agora a consulta Q_2, que possui restrição de duração de tempo em um predicado:

Q_2: Quais trajetórias que possuem o seguinte padrão de movimento: trabalham durante pelo menos 5 unidades de tempo, depois jantam e, logo após, estão em casa?

Sua consulta pode ser representada, em TCTL, da seguinte forma:

$$\mathsf{EF}(\mathsf{E}\, Working \bigcup_{\geq 5} \mathsf{EF}(\mathsf{E}\, Dining \bigcup_{\geq 1} \mathsf{EF}(Home)))$$

A consulta Q_2 representa que, a partir de um estado inicial, existirá, em algum instante de tempo posterior na trajetória (aqui representada como um autômato temporizado), uma configuração na qual o predicado *Working* será verdadeiro por pelo menos 5 unidades de tempo até existir, em um tempo posterior, o predicado *Dining*. O predicado *Dining* deverá ser válido por pelo menos 1 unidade de tempo até existir, em um tempo posterior, o predicado *Home*.

Finalmente, considere a consulta Q_3, que além de possuir restrição de duração de tempo em um predicado, possui também restrição de duração de tempo entre predicados:

Q_3: Quais trajetórias que possuem o seguinte padrão de movimento: realizam uma atividade de trabalho durante pelo menos 2 unidades de tempo, então, depois de no máximo 4 unidades de tempo, estão em casa e, após isto, jantam?

Sua consulta pode ser expressa, em TCTL, da seguinte forma:

$$\mathsf{EF}(\mathsf{E}\,\mathit{Working} \bigcup_{\geq 5} \mathsf{EF}_{\leq 4}(\mathsf{EF}(\mathsf{E}\,\mathit{Home} \bigcup_{\geq 1} \mathsf{EF}(\mathit{Dining}))))$$

Ou seja, a partir de um estado inicial, existirá, em algum instante de tempo posterior na trajetória, uma configuração na qual o predicado *Working* será verdadeiro por pelo menos 5 unidades de tempo até existir, em um tempo posterior a no máximo 4 unidades de tempo, o predicado $Home$. O predicado $Home$ deverá ser válido por pelo menos 1 unidade de tempo até existir, em um tempo posterior, o predicado *Dining*.

4.2.1 Análise de Complexidade

O *autômato temporizado* $\mathscr{A}$ construído para cada trajetória é caracterizado por possuir uma única variável de *clock* e por não ter requisições de reinicialização desta variável em alguma transição. Estes autômatos são chamados de autômatos temporizados de um *clock* (*1C-TAs*). Em (LAROUSSINIE; MARKEY; SCHNOEBELEN, 2004) é apresentado um teorema que mostra que os *1C-TAs* podem ser verificados em tempo polinomial para um fragmento da TCTL, a $TCTL_{\leq,\geq}$ (neste fragmento as restrições do tipo $= c$ são proibidas). Mais especificamente, a verificação de modelos para a $TCTL_{\leq,\geq}$ é da ordem de complexidade de $O(|\Phi|^3 \cdot |L|^2 \cdot |E|^3)$, tal que Φ é a fórmula TCTL utilizada na verificação de modelos, L é o conjunto de estados do *autômato temporizado* e E é o conjunto das arestas do autômato.

Considerando um banco de dados com m trajetórias, nossa abordagem realiza um processo de verificação de modelos para cada trajetória, o que resulta em uma solução com complexidade $O(m \times k)$, onde k é a complexidade de tempo da verificação de modelos em TCTL baseado em *autômato temporizado*. Como nossa abordagem considera 1C-TAs e fórmulas TCTL com restrições $\sim = \{<, \leq, \geq, >\}$, a complexidade de tempo é polinomial, pois a verificação de modelos será realizada para um número limitado m de trajetórias, e como a verificação de modelos é realizada na ordem polinomial de tempo, logo nossa abordagem será polinomial.

Apesar do resultado satisfatório apresentado, soluções práticas em verificação de modelos foram construídas para casos gerais e podem ter complexidade exponencial no pior caso.

4.3 Especificação da Linguagem de Consulta

Visando facilitar o processo de escrita das consultas que expressam padrões de movimento, uma linguagem de consulta é proposta nesta seção. Esta linguagem servirá como um

syntactic sugar para fórmulas TCTL, ou seja, esta linguagem irá simplificar a escrita de padrões de movimento, não exigindo do usuário conhecimento acerca de lógica temporal, assim como de TCTL. Os padrões de movimento são vistos como uma sequência de predicados que são válidos em um determinado intervalo de tempo, ao longo de uma trajetória. Algumas das consultas que devem ser realizadas são:

- (i) quais trajetórias descrevem um objeto móvel que esteve em um determinado local, realizando uma determinada atividade, depois foi conduzido por um determinado meio de transporte, para, depois, chegar a casa;
- (ii) quais trajetórias descrevem um objeto móvel que realizou uma determinada atividade durante 2 horas, por exemplo, depois passou por um determinado local e, após 3 horas, por exemplo, foi trabalhar em um determinado local.

Como forma de facilitar a compreensão da linguagem, optou-se por defini-la através de uma BNF. Em seguida, são descritos cada um dos construtores da linguagem e apresentadas às respectivas fórmulas TCTL que eles representam.

⟨**Moving Pattern**⟩→⟨Episode⟩| ⟨Episode⟩;⟨Moving Pattern⟩| ⟨Episode⟩;[∼ c]⟨Moving Pattern⟩
⟨**Episode**⟩→⟨Expression⟩| (⟨Expression⟩ ∼ c)
⟨**Expression**⟩→⟨predicate⟩| (⟨predicate⟩ *AND* ⟨Expression⟩)| (⟨predicate⟩ *OR* ⟨Expression⟩)
⟨**predicate**⟩→$P \in Predicates$
∼→ representa um operador relacional**,** que pode ser **<, <=, >** ou **>=.**
c→ um número natural $\mathbb{N}$ que representa a quantidade de tempo**.**

Pode-se observar, na BNF, que os principais componentes da linguagem (i.e. Moving Pattern, Event e Expression) são definidos de forma recursiva, proporcionando a construção da consulta através de componentes básicos. Cada um dos componentes é descrito abaixo, nos quais *Predicates* é o conjunto de todas as descrições de predicado da aplicação. Cada predicado pertencente à *Predicates* representa uma proposição ou fórmula atômica da lógica proposicional.

- **Moving Pattern:** representa o padrão de movimento, apresentado na definição 2.2.2, e pode ser representado por um episódio único ou por um episódio seguido de um operador sequencial e depois por um padrão de movimento (*Moving Pattern*). Os operadores sequenciais podem ser ";"ou ";[∼ c]". É possível notar que o padrão de movimento é definido recursivamente. Abaixo segue a descrição de cada forma de se construir um ⟨Moving Pattern⟩, assim como sua representação em TCTL.
  - ⟨Episode⟩ - o padrão de movimento é caracterizado por um único episódio e, em TCTL, a fórmula representativa irá depender de como um episódio é definido (ver o item sobre o componente ⟨Episode⟩);

  - $\langle$Episode$\rangle$;$\langle$Moving Pattern$\rangle$ - este operador sequencial indica que um padrão de movimento ocorrerá posterior à ocorrência de um episódio. Em TCTL, a fórmula representativa será: $\mathsf{EF}(\mathsf{E}\langle\text{Episode}\rangle \bigcup_{\geq 1} \langle\text{Moving Pattern}\rangle)$. Suponha que, através da fórmula $\langle$Episode$\rangle$;$\langle$Moving Pattern$\rangle$, a consulta $ep_1;ep_2$ seja escrita, com ep_1 e ep_2 sendo dois episódios, então o episódio ep_1 ocorreu em um instante de tempo anterior a ocorrência de ep_2;
  - $\langle$Episode$\rangle$;$[\sim c]\langle$Moving Pattern$\rangle$ - este operador sequencial indica que um padrão de movimento ocorrerá posterior à ocorrência de um episódio, tal que uma determinada duração de tempo (c unidades de tempo) tenha sido transcorrida entre a ocorrência do episódio e a ocorrência do padrão de movimento. Em TCTL, a representação é a seguinte: $\mathsf{EF}(\mathsf{E}\langle\text{Episode}\rangle \bigcup_{\sim c} \langle\text{Moving Pattern}\rangle)$. Como exemplo, considere a fórmula $ep_1;[\geq 2]ep_2$, com ep_1 e ep_2 sendo dois episódios, então o episódio ep_2 ocorreu em um instante ou período de tempo depois de ter transcorrido, ao menos, 2 unidades de tempo da ocorrência do episódio ep_1. Para os outros operadores relacionais ($\{\leq, <, >, \geq\}$) a interpretação é semelhante, apenas com a diferença explicitada pelos respectivos operadores.

- **Episode:** refere-se ao episódio definido no modelo de dados e pode ser representado por uma expressão (*Expression*) ou por uma expressão seguida de uma restrição de duração ($\sim c$). O componente *Expression* define como um episódio pode ser representado. Abaixo, são apresentados os construtores para *Episode*:

  - $\langle$Expression$\rangle$ - uma expressão, que será definida posteriormente. Em TCTL, a verificação da ocorrência de uma expressão será representada por: $\mathsf{EF}\langle\text{Expression}\rangle$;
  - ($\langle$Expression$\rangle \sim c$) - uma expressão com uma restrição de duração ($\sim c$), indicando que os predicados da expressão serão válidos durante uma determinada quantidade de tempo. As restrições apresentadas são as que seguem: $\leq c$ - duração de no máximo c instantes de tempo, $< c$ - a duração não pode chegar a c instantes de tempo, $\geq c$ - duração de no mínimo c instantes de tempo e $> c$ - duração maior do que c instantes de tempo. A representação em TCTL é a seguinte: $\mathsf{EF}(\mathsf{E}\langle\text{Expression}\rangle \bigcup_{\sim c} \top)$, ou seja, *Expression* irá valer, em algum momento no futuro, até que qualquer outra coisa seja válida ($\top$), respeitando a restrição de duração de tempo ($\sim c$).

- **Expression:** define quais os predicados podem ser válidos em um determinado episódio. A expressão é definida por um predicado ou por um predicado seguido por um operador lógico e depois por uma expressão. Ou seja, uma expressão é definida somente com o uso de predicados, suas conjunções (*AND*) e/ou disjunções (*OR*). A semântica deste componente é definida da seguinte forma, para cada construtor (para todos os casos considere $P \in Predicates$):

  - $\langle$predicate$\rangle$ - para este caso pode ocorrer a descrição P, que significa que existe um instante ou período de tempo no qual o predicado P vale. Exemplos de predicados: $Home$, $Working$, etc.;

- $(\langle \text{predicate} \rangle \, AND \, \langle \text{Expression} \rangle)$ - significa que existe um instante ou período de tempo no qual P e os predicados de $\langle \text{Expression} \rangle$ valem. Como exemplo, tem-se: $(Home \, AND \, \langle \text{Expression} \rangle)$;
- $(\langle \text{predicate} \rangle \, OR \, \langle \text{Expression} \rangle)$ - significa que existe um instante ou período de tempo no qual ou o predicado P ou os predicados de $\langle \text{Expression} \rangle$ valem neste instante ou período. Como exemplo, tem-se: $(Home \, OR \, \langle \text{Expression} \rangle)$.

Após a definição de cada construtor da linguagem a Tabela 4.1 foi construída visando resumir as quais fórmulas TCTL os componentes da linguagem representam.

Componente da linguagem	**TCTL Fórmula**
$\langle \text{predicate} \rangle$	P
$(\langle \text{predicate} \rangle \, AND \, \langle \text{Expression} \rangle)$	$(P \, AND \, \langle \text{Expression} \rangle)$
$(\langle \text{predicate} \rangle \, OR \, \langle \text{Expression} \rangle)$	$(P \, OR \, \langle \text{Expression} \rangle)$
$\langle \text{Expression} \rangle$	$\mathsf{EF} \, \langle \text{Expression} \rangle$
$(\langle \text{Expression} \rangle \sim c)$	$\mathsf{EF}\,(\mathsf{E}\,\langle \text{Expression} \rangle \bigcup_{\sim c} \top)$
$\langle \text{Episode} \rangle$	$\langle \text{Expression} \rangle \mid (\langle \text{Expression} \rangle \sim c)$
$\langle \text{Episode} \rangle ; \langle \text{Moving Pattern} \rangle$	$\mathsf{EF}(\mathsf{E}\,\langle \text{Episode} \rangle \bigcup_{\geq 1} \langle \text{Moving Pattern} \rangle)$
$\langle \text{Episode} \rangle ; [\sim c] \langle \text{Moving Pattern} \rangle$	$\mathsf{EF}(\mathsf{E}\,\langle \text{Episode} \rangle \bigcup_{\sim c} \langle \text{Moving Pattern} \rangle)$

Tabela 4.1: Componentes e suas fórmulas TCTL equivalentes.

4.4 Exemplos de Consultas

Definidos os construtores e os termos da BNF, pode-se então construir as consultas mencionadas na seção 2.2 de acordo com a linguagem aqui definida. Seja, então, a consulta Q_1 apresentada abaixo.

Q_1: Quais as trajetórias possuem o seguinte padrão de movimento: partem de casa, depois vão para a universidade e, então, retornam para casa?

Esta pode ser representada, na linguagem de consultas, da seguinte forma:

(Home);(At_University);(Home)

Considere agora a consulta Q_2, que possui restrição de duração de tempo em um predicado:

Q_2: Quais trajetórias que possuem o seguinte padrão de movimento: trabalham durante pelo menos 5 unidades de tempo, depois jantam e, logo após, estão em casa?

Sua consulta pode ser representada, na linguagem, da seguinte forma:

(Working>=5);(Dining);(Home)

Finalmente, considere a consulta Q_3, que além de possuir restrição de duração de tempo em um predicado, possui também restrição de duração de tempo entre predicados:

Q_3: Quais trajetórias que possuem o seguinte padrão de movimento: realizam uma atividade de trabalho durante pelo menos 2 unidades de tempo, então, depois de no máximo 4 unidades de tempo, estão em casa e, após isto, jantam?

Sua consulta pode ser representada, na linguagem, da seguinte forma:

(Working>=2);[<=4](Home);(Dining)

4.5 Conclusão

Apresentamos neste capítulo uma solução para problema de expressar consultas sobre padrões de movimento e processar tais consultas no sentido de recuperar trajetórias que correspondam a tais padrões. Para resolver este problema, primeiramente foi estudado Lógicas Temporais com a finalidade de definir, formalmente, os operadores da linguagem. A partir de então, verificou-se que o problema poderia ser interpretado com um problema de verificação de modelos, no qual cada trajetória seria um modelo e a consulta seria uma propriedade a ser verificada no modelo. Foi verificada que a solução proposta para a realização de consultas poderá ser executada em tempo polinomial. Ao final do capítulo, foi proposta uma linguagem de consulta que expressa, de uma forma mais natural para o usuário, as consultas expressas em TCTL, não exigindo conhecimentos acerca de lógica temporal. Assim, a linguagem proposta servirá apenas como um *syntactic sugar* para fórmulas expressas em TCTL.

5 DESENVOLVIMENTO DE PROTÓTIPO E EXPERIMENTOS

Neste capítulo apresentamos um protótipo que possibilita a construção de uma consulta expressa na nossa linguagem e a sua avaliação através da realização de verificações de modelos utilizando a API da ferramenta Uppaal. A abordagem de avaliação de consultas utilizada pelo protótipo foi apresentada de forma detalhada na Seção 5.3. Ao final, apresentamos alguns resultados considerando a realização de consultas em um conjunto de dados de trajetórias. Foi analisada a duração da consulta considerando o tamanho da base de dados e a quantidade de predicados da consulta.

5.1 Metodologia

Alguns experimentos foram propostos, visando demonstrar que a abordagem apresentada nesta dissertação é escalável quanto ao tamanho do banco de dados, quanto ao número de predicados presentes no padrão de movimento e quanto ao número de predicados com restrições temporais presentes em um consulta. Os detalhes de cada um dos experimentos podem ser verificados a seguir:

- **Escalabilidade quanto ao número de trajetórias:** Neste experimento será utilizada uma mesma consulta que expressa um dado padrão de movimento para tamanhos de banco de dados diferentes. Este experimento visa demonstrar que a proposta apresentada nesta dissertação é viável para grande quantidade de dados. Os testes devem ser realizados sobre banco de dados contendo as seguintes quantidades de trajetórias semânticas: 10, 100, 1000, 10000. Neste caso optou-se por uma escala exponencial de crescimento da quantidade de trajetórias, visando possibilitar uma leitura mais ampla dos resultados.

- **Escalabilidade quanto à quantidade de predicados:** Este experimento consiste na realização de consultas simples, nas quais sempre irão verificar se uma sequência de predicados ocorre em uma determinada ordem para uma quantidade fixa de trajetórias semânticas no banco. Assim, para cada tamanho do banco de dados (i.e. 10, 100, 1000, 10000 e 100000), será verificada a quantidade de tempo gasta quando a quantidade de predicados presentes em um padrão de movimento será variada, assumindo os seguintes valores: 2, 4, 8, 16 e 32. Este experimento visa demonstrar que, por mais que a quantidade de predicados presentes no padrão de movimento aumente, a consulta poderá ser processada em tempo proporcional a quantidade destes predicados.

- **Escalabilidade quanto à quantidade de predicados com restrição temporal:** Este experimento consiste na realização de uma consulta com 32 predicados na qual a quantidade de predicados com restrições temporais será variada a uma taxa de crescimento exponencial em 2. A quantidade de predicados temporais assumirá os seguintes valores: 2, 4, 8, 16 e 32. Este experimento visa demonstrar o impacto, na duração de tempo de consulta, quando a quantidade de predicados com restrição temporal aumenta no padrão de movimento descrito pela consulta.

5.2 Escolha do Verificador de Modelos

Após a definição dos experimentos a serem realizados para comprovar a eficiência da abordagem proposta no Capítulo 4, alguns verificadores de modelos foram analisados objetivando a implementação de um protótipo para a realização de testes.

O primeiro verificador de modelos a ser analisado foi o *NuSMV* (CIMATTI et al., 1999), um verificador simbólico de modelos bastante difundido pela comunidade acadêmica. Entretanto, esta ferramenta tem como objetivo principal realizar a verificação de modelos para as lógicas CTL e LTL, não permitindo expressões de restrições temporais nos operadores modais, além de não permitir a definição de modelos com restrições de tempo, como o utilizado neste trabalho, baseado na ideia de autômato temporizado. A partir de então, buscou-se por verificadores de modelos voltados para a lógica TCTL, esta que é utilizada para descrever os operadores da linguagem.

Outro verificador de modelos analisado foi o *TSMV* (MARKEY; SCHNOEBELEN, 2004), que é uma extensão do NuSMV para sistemas temporizados que trabalha com a lógica TCTL. Entretanto, foi verificado que o TSMV não está mais sendo mantido e, por este motivo, decidiu-se por não utilizá-lo.

Ao final, optou-se por utilizar o Uppaal, um verificador de modelos para lógica TCTL que permite a modelagem, simulação e verificação de sistemas em tempo real. Esta escolha deve-se ao fato de o Uppaal lidar com um pequeno conjunto da TCTL e por possibilitar a modelagem em autômatos temporizados, possibilitando que a modelagem de trajetórias, proposta por este trabalho, seja implementada. O Uppaal foi desenvolvido por uma colaboração entre o *Department of Information Technology* da Universidade Uppsala, Suécia, e o *Department of Computer Science* da Universidade de Aalborg, Dinamarca. A ferramenta lida com um pequeno subconjunto da lógica TCTL, permitindo a construção de grandes modelos por modelar o sistema como uma rede de diversos *autômato temporizados* em paralelo (BEHRMANN; DAVID; LARSEN, 2006). A verificação de modelos no Uppaal é baseada na verificação da alcançabilidade de um estado em um *autômato temporizado* (LARSEN; PETTERSSON; YI, 1997), devido a esta propriedade a ferramenta realiza a verificação de modelos, para o caso geral, na ordem de complexidade PSPACE-completo (BOUYER; LAROUSSINIE, 2010).

Embora o Uppaal possa ser menos eficiente, do ponto de vista teórico, em relação à abordagem proposta na Seção 4.2, decidiu-se utilizá-lo, pois construir um verificador de modelos é um trabalho muito oneroso, além de o Uppaal ser amplamente utilizado e ter diversas otimizações, resultado de 15 anos de ativo desenvolvimento. Como a solução proposta lida com autômatos finitos para cada trajetória semântica e um pequeno conjunto da lógica TCTL é usado para a tradução da linguagem de consultas (i.e. operadores $EF_{\sim c}\varphi$ e $E(\varphi\ U_{\sim c}\ \psi)$), espera-se que a utilização desta ferramenta seja eficiente na prática.

No Uppaal, a verificação de modelos é baseada na teoria do *autômato temporizado* (ALUR; DILL, 1990), seguindo a Definição 4.1.1 exposta no Capítulo 4, utilizando-se de características adicionais, como um conjunto de ações A, **estados de urgência** e **estados *commited*** (BEHRMANN; DAVID; LARSEN, 2006). Para uma melhor compreensão do conceito de autô-

mato temporizado utilizado pelo Uppaal, uma definição é apresentada a seguir. Esta definição é semelhante à definição 4.1.1, no entanto possui as particularidades do autômato temporizado utilizado pelo Uppaal.

Definição 5.2.1 (Autômato Temporizado - Uppaal) *Um autômato temporizado no Uppaal é uma tupla* $(L, l_0, \mathscr{C}, A, E, I)$*, na qual* L *é o conjunto de estados;* l_0 *é o estado inicial;* $\mathscr{C}$ *é um conjunto de variáveis de clock;* A *é um conjunto de ações;* $E \subseteq L \times A \times \mathscr{B}(\mathscr{C}) \times 2^{\mathscr{C}} \times L$ *é um conjunto de arestas entre os estados com uma anotação* $a \in A$*, restrições de guarda do tipo* $x \sim c \in \mathscr{B}(\mathscr{C})$ *e um conjunto de variáveis de clock a serem reinicializadas representado por um conjunto pertencente a* $2^{\mathscr{C}}$*; e* $I : L \to \mathscr{B}(\mathscr{C})$ *atribui invariantes aos estados.*

Com relação aos estados adicionais de urgência e *commited*, cada um possui uma definição distinta. O estado de **urgência** (representado com um U) é equivalente a adicionar-se um *clock* extra x ao modelo, que é reinicializado em todas as transições de entrada do estado e tem, como invariante do estado, o valor $x \leq 0$, ou seja, não é permitido que o tempo evolua enquanto se está em um estado urgente. O estado *commited* (representado com um C) não transcorre o tempo e requer que a próxima ação envolva uma transição cuja fonte é o estado *commited*. Este estado permite que uma determinada propriedade seja verificada para os mesmos valores das variáveis de *clock* tanto para as transições de chegada, como para as transições de saída. Seu uso será explicado com mais detalhes posteriormente.

No Uppaal, as transições podem ter uma ação, representada pelos canais de sincronização. Desta forma, quando dois autômatos são sincronizados no canal "a", por exemplo, significa que a transição marcada com "a!", em um autômato, ocorre simultaneamente com a transição marcada com "a?" do outro autômato. O marcador "!" representa o componente que toma a "iniciativa", enquanto que o marcador "?" representa o componente "passivo". A Figura 5.1 apresenta o processamento simultâneo que ocorre no Uppaal para duas transições sincronizadas em um mesmo canal, sendo uma transição marcada com "!" e a outra com "?".

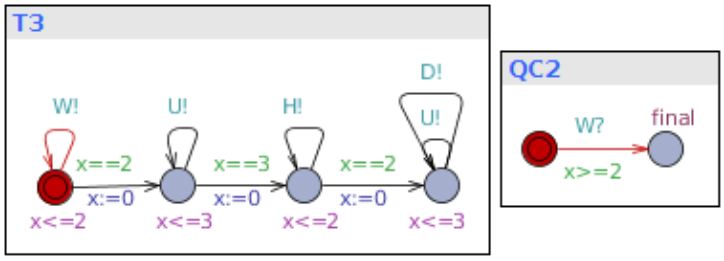


Figura 5.1: A transição $W!$ do autômato $T3$ ocorre ao mesmo tempo em que a transição $W?$ do autômato $QC2$ acontece, como pode ser observado pela transição em vermelho.

O Uppaal não lida com a TCTL completa, especialmente os casos de restrições de tempo e os casos de aninhamento de operadores modais (e.g. $\mathsf{E}(\varphi \bigcup_{\sim c} \psi)$ e $\mathsf{EF}_{\sim c}\mathsf{E}(\varphi \bigcup \psi)$). As fórmulas aceitas no Uppaal são: $\mathsf{A}[]\varphi$, $\mathsf{A}<>\varphi$, $\mathsf{E}<>\varphi$, $\mathsf{E}[]\varphi$ e $\psi \rightsquigarrow \varphi$[1]. Para cada uma destas fórmulas, φ e ψ são fórmulas de estado. Uma fórmula de estado é uma expressão que

[1] $\psi \rightsquigarrow \varphi$ equivale a $\mathsf{A}\Box(\varphi \Rightarrow \mathsf{A}\Diamond\psi)$ em TCTL

pode ser avaliada para um estado sem precisar analisar o comportamento de todo o modelo. Um exemplo de uma fórmula de estado é $Q.final$, na qual Q é um modelo (e.g. um *autômato temporizado*) e $final$ é um estado deste modelo (BEHRMANN; DAVID; LARSEN, 2006)x.

5.3 Tradução da Solução para o Uppaal

Como apresentado no Capítulo 2, uma trajetória semântica é representada como uma sequência de episódios, que serão representadas, cada uma, por um *autômato temporizado*, com somente um *clock x*, no qual cada episódio $ep = (start, end, \Psi)$ será representado por um estado com $x <= end - start + 1$ como invariante, ou seja, cada episódio deve ter a duração de no mínimo uma unidade de tempo. Como o Uppaal não possui, para os autômatos temporizados representados nesta ferramenta, a função de valoração $V : L \rightarrow 2^{AP}$, que indica as fórmulas atômicas válidas em um dado estado, será necessário o uso das anotações A para a representação dos predicados válidos em determinados intervalos de tempo em uma dada trajetória. Assim, cada predicado $P \in \Psi$, em um episódio, será um canal no Uppaal e será representado, no *autômato temporizado*, como uma auto-transição com o marcador de sincronização $P!$ para o estado correspondente ao episódio em questão. Desta forma, quando a transição marcada com $P!$ for executada, ela irá disparar a execução de alguma transição $P?$ que possa existir. As transições entre os estados seguem a ordem de sucessão entre os episódios, para dois episódios sucessivos $ep_1 = (start_1, end_1, \Psi_1)$ e $ep_2 = (start_2, end_2, \Psi_2)$, constrói-se uma transição, do estado que representa o primeiro episódio, para o estado que representa o segundo, tal que a restrição de guarda g seja $x = end_1 - start_1 + 1$, ou seja, o último valor que satisfaz a restrição invariante do estado referente ao episódio ep_1. Em cada uma das transições entre os estados, a variável de *clock x* será reinicializada para 0. Para facilitar a compreensão, foi construído o Algoritmo 2 para realização da construção do *autômato temporizado* referente a uma dada trajetória semântica.

Visando exemplificar como seriam os *autômatos temporizados* referentes a algumas trajetórias, sejam as 3 trajetórias apresentadas abaixo:

- **Trajetória 1:** $\{(1,3,[Home]), (3,5,[Working, At\_University]), (5,6,[Home])\}$;
- **Trajetória 2:** $\{(1,5,[Working]), (5,6,[Home]), (6,9,[Home, Dining])\}$.
- **Trajetória 3:** $\{(1,2,[Working]), (2,4,[At\_University]), (4,5,[Home]), (5,7, [At\_University, Dining])\}$;

As trajetórias 1, 2 e 3 são respectivamente representadas, no Uppaal, de acordo com o Algoritmo 2, pelos *autômatos temporizados* das figuras 5.2a, 5.2b e 5.2c. Nas figuras, o nó com um círculo duplo significa o estado inicial. Para as figuras presentes na Figura 5.2, considere H - $Home$, W - $Working$, U - $At\_University$ e D - $Dining$.

É possível notar que a construção dos *autômatos temporizados* das trajetórias semânticas, de acordo com o Algoritmo 2, resulta na exclusão da informação sobre a exatidão da

Algoritmo 2: Construção do *autômato temporizado* da trajetória no Uppaal

Entrada: Uma lista EL ordenada de episódios $ep_i = (start_i, end_i, \Psi_i)$, tal que $1 \leq i \leq n$, pertencente a uma dada trajetória semântica TS, cuja a quantidade de episódios é n

Saída: Um *autômato temporizado* do Uppaal M.

```
1  duração ⟵ 0;
2  para ep ∈ E faça
3      Crie um estado l;
4      se ep é o primeiro então
5          Faça l ser inicial;
6      senão
7          l_predecessor ← o estado referente predecessor(estado);
8          Crie uma transição trans de l_predecessor para l com guarda x == duração e
           atualização x := 0;
           // Ou seja, uma transição do estado correspondente ao episódio
              predecessor para o episódio atual
9          Inclua trans no autômato temporizado M;
10     fim se
11     duração ⟵ ep.end − ep.start + 1;
12     Inclua como invariante do estado estado o valor x <= duração;
13     para p ∈ ep.Ψ faça
14         Crie uma auto-transição trans em l com o rótulo p!;
15         Inclua trans no autômato temporizado M;
16     fim para
17     Inclua l no autômato temporizado M;
18 fim para
19 retorna autômato temporizado M;
```

ocorrência de um determinado episódio. Na trajetória 1, por exemplo, não podemos identificar no *autômato temporizado* que o segundo episódio da trajetória ocorre entre os tempo 4 e 6. A informação que consta no autômato é que a duração deste episódio foi de 3 instantes de tempo. Mais estudos serão necessários para a inclusão desta informação, visando possibilitar a realização de consultas que incluam restrições temporais sobre o instante de tempo de um determinado acontecimento, como por exemplo: *estava trabalhando às 5 horas da tarde*.

A respeito das consultas expressas pela linguagem, estas não poderão ser escritas como uma fórmula lógica TCTL, como apresentado na solução exposta no Capítulo 4. Isto se deve ao fato de o Uppaal não permitir o aninhamento de operadores modais, como explicado no início da Seção 5.2. Neste caso, uma possibilidade para representar a consulta é através de sua tradução para um autômato temporizado, o qual irá representar um padrão de movimento descrito como uma sequência de predicados (com ou sem restrições de duração) como uma sequência de transições cujas ações se referem aos predicados da sequência. Por exemplo, seja a consulta Q_1 abaixo, cuja representação na linguagem proposta é $(Home);(At\_University);(Home)$:

Q_1: Quais as trajetórias possuem o seguinte padrão de movimento: partem de

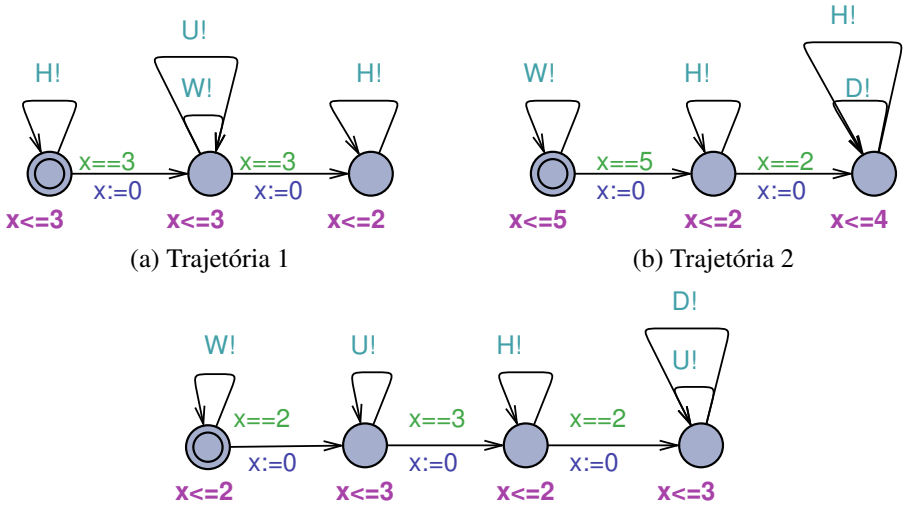


Figura 5.2: Trajetórias modeladas como um *autômato temporizado* no Uppaal

casa, depois vão para a universidade e, então, retornam para casa?

Esta consulta pode ser traduzida para o *autômato temporizado* no Uppaal representado na Figura 5.3, onde H representa o predicado $Home$ e U representa o predicado $At\_University$.

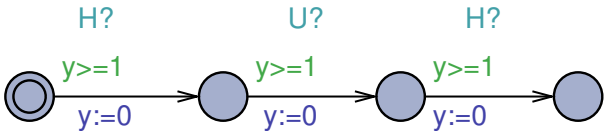


Figura 5.3: Representação da consulta Q_1 como um *autômato temporizado* no Uppaal com a restrição que força ao sistema transcorrer em tempo discreto.

Pode-se notar que as ações do autômato de consulta são passivas, devido ao marcador "?", assim, estas transições só serão realizadas quando alguma transição do autômato de trajetória, que possua a mesma ação com o marcador "!", for realizada. Desta forma, se o autômato de consultas chegar ao estado *final*, então é porque o autômato da trajetória possui a sequência de predicados na mesma ordem em que estes aparecem na consulta. É importante enfatizar, também, que o sequenciamento de predicados através do operador ";", por exemplo, $(Home);(At\_University)$, exige que exista um instante ou período de tempo no qual $(Home)$ vale e $(At\_University)$ vale em um outro instante ou período posterior ao que vale $(Home)$. Para a formalização disto no Uppaal, foi definido utilizar o conjunto dos naturais ($\mathbb{N}$) como domínio de tempo, ou seja, o domínio de valores atribuídos às variáveis de clock. Como esclarecido na Seção 5.2, o domínio de tempo do Uppaal é o conjunto dos números reais ($\mathbb{R}$), por este motivo, deve-se forçar ao Uppaal a lidar com tempo discreto. Uma forma de se fazer isto é adicionar uma variável de *clock* "y" no autômato de consulta, que terá seu valor atualizado para 0 e uma restrição de guarda $y \geq 1$ para cada transição do autômato. Desta forma, o Uppaal não irá lidar com os números reais, transcorrendo, assim, pelo menos 1 unidade de tempo quando ocorrer uma transição no autômato temporizado de consulta.

De forma geral, a consulta no Uppaal será um *autômato temporizado*, com um clock adicional y e com suas restrições referentes ao problema do tempo discreto exposto anteriormente. Cada episódio da linguagem de consultas será traduzido para um *autômato temporizado* e a consulta toda será a fusão de todos os autômatos, ou seja, o estado final de um autômato será o estado inicial do autômato seguinte, seguindo a ordem de sucessão dos episódios. Neste autômato as transições terão marcadores de sincronização do tipo $X?$, onde X é um predicado e "?" indica que esta transição estará esperando a ação de uma transição $X!$. Para o autômato de consulta, o estado final será nomeado como "*final*", visando possibilitar a realização da verificação de modelos que corresponde a analisar se uma trajetória satisfaz a consulta formulada na linguagem proposta.

Como forma de entender como ocorre a tradução de uma consulta escrita na linguagem proposta na Seção 4.3, foi desenvolvido o Algoritmo 3, apresentado a seguir.

Algoritmo 3: Construção do *autômato temporizado* referente a uma consulta expressa pela linguagem definida neste trabalho

Entrada: Uma consulta Q escrita de acordo com a linguagem proposta
Saída: Um *autômato temporizado* do Uppaal $\mathscr{A}$.

```
1  primeiro_episodio ⟵ pega_primeiro_episodio();
2  automato_ant ⟵ contruir_automato_episodio(primeiro_episodio);
3  para ep ∈ Q faça
4  |   estado ⟵ cria_estado();
5  |   Expression ⟵ pega_primeira_expression(ep);
6  |   automato ⟵ contruir_automato_episodio(Expression, tem_duracao(ep),
   |   estado);
7  |   𝒜 ⟵ concatena(automato_ant, automato);
8  |   automato_ant ⟵ automato;
9  fim para
10 retorna autômato temporizado 𝒜;
```

É possível notar que a construção dos *autômatos temporizados* referentes as consultas construídas, de acordo com o Algoritmo 3, acontece através da construção de um autômato menor para cada episódio e da concatenação destes autômatos. A construção do autômato temporizado referente ao episódio é realizada através do Algoritmo 4 e consiste em construir um autômato representativo das expressões, descritas como disjunções e/ou conjunções de predicados, juntamente com uma restrição de duração de tempo que a expressão possa ter. Já com relação à concatenação destes autômatos, esta concatenação consiste unir o último estado do autômato referente a um episódio com o primeiro estado do autômato referente ao próximo estado. Desta forma, a concatenação dos autômatos, como apresentado no Algoritmo 3, segue a ordem temporal dos episódios em uma trajetória. Ao final, o autômato temporizado, resultante da concatenação dos autômatos referentes aos episódios, é retornado. Para entender como acontece a concatenação dos autômatos, foi escrito o Algoritmo 5, que pode ser visto a seguir.

Pode-se observar no Algoritmo 4 que a construção do autômato temporizado de um episódio específico irá depender dos operadores lógicos presentes em sua expressão. A partir de uma expressão de entrada ($\langle$Expression$\rangle$), de um indicador de presença de restrição de

Algoritmo 4: construir_automato_episodio - Construção do *autômato temporizado* referente a uma consulta expressa pela linguagem definida neste trabalho

Entrada: Uma expressão (⟨Expression⟩) representada por Expression; uma variável booleana `tem_restricao_duracao` indicando se o episódio possui restrição de duração; um estado souce, que indica o primeiro estado do autômato.
Saída: Um *autômato temporizado* do Uppaal $\mathscr{A}$.

```
1  estado ⟵ souce;
2  transicao ⟵ cria_transicao();
3  transicao.setSource(estado);
4  predicado ⟵ captura_primeiro_predicado(Expression);
5  transicao.setAction(predicado ?);
6  se tem_restricao_duracao então
7  |  transicao.setGuard("∼ c");
8  fim se
9  se Expression = predicate AND ⟨Expression⟩ então
10 |  estado ⟵ cria_estado_commited();
11 |  transicao.setTarget(estado);
12 |  contruir_automato_episodio(Expression,tem_restricao_duracao,
   |  estado);
13 senão se Expression = predicate OR ⟨Expression⟩ então
14 |  transicoesAntesOR ⟵ transicao;
15 |  contruir_automato_episodio(Expression,tem_restricao_duracao,
   |  estado);
16 senão
17 |  estado ⟵ cria_estado(); estado.setName(final);
18 |  transicao.setTarget(estado);
19 fim se
20 para transicao ∈ transicoesAntesOR faça
21 |  transicao.setTarget(estado);
22 fim para
23 retorna autômato temporizado 𝒜 representado pelos estados e transições criados;
```

duração e de um estado inicialmente criado, o algoritmo cria uma transição para cada predicado pertencente à expressão, de forma recursiva. O processo de criação das transições pode ser observada nas linhas 2 a 5, onde na linha 2 a transição é criada, na linha 3 é definido o estado de origem da transição (método *setSource*), na linha 4 é capturado o primeiro predicado da expressão (i.e. o predicado mais à esquerda da expressão) e na linha 5 a ação da transição terá o marcador de sincronização "*predicate*?", indicando que a transição só será processada quando uma transição com o marcador "*predicate*!"for processada no autômato referente à trajetória. Se o episódio que está sendo processado pelo algoritmo possuir uma restrição de duração, representada por $\sim c$, a mesma será incluída na transição através da inclusão de uma restrição de guarda representando a restrição, linha 7.

Após a criação da transição para um predicado pertencente à expressão de um episódio, o algoritmo fará chamadas recursivas a ele próprio, baseado em como a expressão constituída. Se a expressão tem o formato "*predicate AND* ⟨Episode⟩", será construído um estado

commited para o qual a transição criada irá apontar (linhas 10 e 11) e então a chamada recursiva é realizada (linha 12). Ou seja, para cada operador lógico *AND* será construído um estado *commited* que irá representar este operador. Isto ocorre porque todas as transições que chegam ao estado *commited* e que saem deste estado, devem ocorrer para os mesmos valores de clock, não transcorrendo tempo. Assim, as transições que chegam ao estado *commited* e que partem dele só serão processadas quando transições, com os mesmos predicados inclusos nas transições do estado *commited* (no entanto com o marcador "!"), forem auto-transições (i.e. ocorrerem no mesmo tempo) no autômato referente à trajetória. Para entender a construção do autômato para episódios que tenha uma expressão do tipo "$predicate\,AND\,\langle\text{Episode}\rangle$", verifique a Figura 5.4.

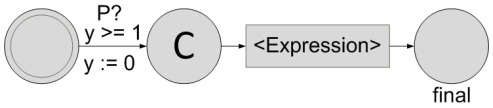


Figura 5.4: Quando um $\langle\text{Expression}\rangle$ for $\langle\text{predicate}\rangle\,AND\,\langle\text{Expression}\rangle$. *P* é um predicado.

De forma simples, a Figura 5.4 explicita que a construção de um autômato temporizado referente a um episódio cuja expressão seja $\langle\text{predicate}\rangle\,AND\,\langle\text{Expression}\rangle$ irá constituir de uma transição com o predicado (*P*?), juntamente como o marcador de transição passiva "?", para um estado *commited* e de uma transição, construída de forma recursiva para o conteúdo de $\langle\text{Expression}\rangle$, do estado *commited* para um estado nomeado como **final**. É importante frisar que a primeira transição, para a expressão abordada nesta figura, irá conter a variável de *clock* adicional *y*, juntamente com sua restrição de guarda $y >= 1$ e a reinicialização da variável $y := 0$. Estas restrições estarão presentes somente na primeira transição, indicando que o episódio deve valer por pelo menos 1 unidade de tempo.

Para o caso da expressão do episódio ter o formato "$predicate\,OR\,\langle\text{Episode}\rangle$", a transição criada no início do Algoritmo 4 é incluída em uma lista de transições ($transitionsBeforeOr$ na linha 14) e a chamada recursiva é realizada (linha 15). Esta lista de transições será processada ao final do algoritmo, pois, para cada operador *OR* encontrado, o estado fonte da transição criada para o predicado antecessor ao operador *OR* será o mesmo estado fonte para a transição que será criada para o predicado posterior ao operador *OR*. Para entender melhor, a Figura 5.5 é apresentada.

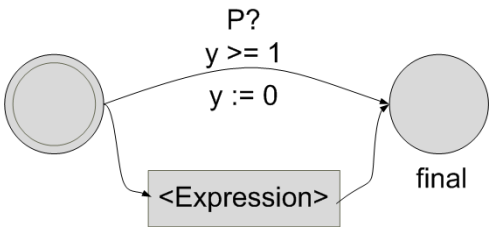


Figura 5.5: Quando um $\langle\text{Expression}\rangle$ for $\langle\text{predicate}\rangle\,OR\,\langle\text{Expression}\rangle$. *P* é um predicado.

De forma simples, a Figura 5.5 explicita que a construção de um autômato temporizado referente a um episódio cuja expressão seja $\langle\text{predicate}\rangle\,OR\,\langle\text{Expression}\rangle$ irá constituir de uma transição com o predicado (*P*?), juntamente como o marcador de transição passiva "?", para um estado nomeado como **final** e de uma transição, construída de forma recursiva para

o conteúdo de ⟨Expression⟩, do mesmo estado fonte da transição *P*? para um estado nomeado como **final**. Esta construção representa uma disjunção, ou seja, a transição a ser tomada pode ser qualquer uma entre *P*? e as transições do autômato referente à ⟨Expression⟩. A tomada de decisão de qual transição será processada irá depender da transição ativa (e.g. *P*!) a ser processada pelo autômato referente à trajetória. É importante frisar que a transição referente ao predicado (i.e. a transição *P*?) irá conter a variável de *clock* adicional *y*, juntamente com sua restrição de guarda $y >= 1$ e a reinicialização da variável $y := 0$. Estas restrições, que indicam que o episódio deve valer por pelo menos 1 unidade de tempo, estarão presentes em transições do autômato temporizado referente ⟨Expression⟩, de acordo com o seu formato, como explicado a seguir:

- ⟨predicate⟩ - a transição referente à ⟨predicate⟩ teria as restrições apresentadas anteriormente;
- ⟨predicate⟩ *OR* ⟨Expression⟩ - a transição referente à ⟨predicate⟩ teria as restrições apresentadas anteriormente e as transições referentes a ⟨Expression⟩ dependeriam de como é seu formato;
- ⟨predicate⟩ *AND* ⟨Expression⟩ - a transição referente à ⟨predicate⟩ teria as restrições e as transições referentes a ⟨Expression⟩ não teriam.

Após as explicações sobre o algoritmo de construção do autômato temporizado referente ao episódio, é apresentado a seguir o algoritmo (Algoritmo 5) referente à concatenação dos autômatos referentes a cada episódio da consulta, visando construir um único autômato temporizado representante da consulta escrita na linguagem proposta.

Algoritmo 5: concatena - Construção do *autômato temporizado* referente a uma consulta expressa pela linguagem definida neste trabalho

Entrada: O autômato referente ao primeiro episódio a ser concatenado automato_ant e o autômato referente ao episódio sucessor automato.
Saída: Um *autômato temporizado* do Uppaal $\mathscr{A}$.

1 **para** *Toda* transicao *que aponta para o estado "final"do autômato* automato_ant **faça**
2 **se** *o operador de sequencial* ;$[\sim c]$ *ocorrer entre os episódios representativos de* automato_ant *e do* automato **então**
3 Adicionar restrições de guarda $y \sim c$ em automato;
4 **fim se**
5 Faça a transição apontar para o primeiro estado de automato;
6 **fim para**
7 **retorna** o *autômato temporizado* $\mathscr{A}$, representado pelo autômato automato_ant, resultado da concatenação.;

É possível observar que a concatenação exposta no Algoritmo 5 consiste em fazer todas as transições, que apontam para o último estado do primeiro autômato temporizado, apontarem para o primeiro estado do segundo autômato. Antes de realizar a concatenação, é

verificado se a restrição de duração entre os episódios (i.e. $;[\sim c]$) está presente. Se a verificação é verdadeira, então as restrições de guarda referentes à variável de *clock* adicional y serão atualizadas no autômato referente à segunda entrada do algoritmo (i.e. $automato$) para $y \sim c$, indicando que de um episódio para o outro a restrição temporal, entre episódios, deverá satisfazer $y \sim c$. Ao final, o autômato referente a concatenação dos dois autômatos temporizados é retornado.

Para verificar se uma dada trajetória semântica t satisfaz a uma dada consulta q, iremos realizar uma verificação de modelo no Uppaal no qual teremos como entrada um sistema constituído do autômato representativo da trajetória t e da consulta q, e de uma fórmula lógica expressa em TCTL. A fórmula TCTL escolhida será uma que representa a propriedade de alcançabilidade de um estado, pois uma forma de descobrir se uma trajetória satisfaz uma consulta é verificar se o estado final do autômato de consulta é atingido, ou seja, se todo o autômato de consulta é processado juntamente com o autômato da trajetória. Neste caso, a consulta será da expressa por $E <> Q.final$, em outras palavras, se o estado nomeado como $final$ é alcançável no autômato Q (i.e. o autômato representativo de uma consulta q).

5.4 Exemplos de Consultas no Uppaal

Definidos os autômatos de cada componente da linguagem, pode-se então construir as consultas mencionadas na seção 2.2 de acordo com a tradução definida na Seção 5.3. Os exemplos iniciam-se da consulta Q_2, pois a consulta Q_1 já foi apresentada em forma de autômato na Seção 5.3. Considere a consulta Q_2 apresentada a seguir, o autômato referente a esta consulta é apresentado na Figura 5.6.

> *Q_2: Quais trajetórias que possuem o seguinte padrão de movimento: trabalham durante pelo menos 5 unidades de tempo, depois jantam e, logo após, estão em casa?*

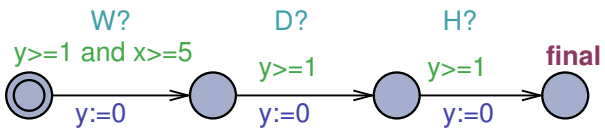


Figura 5.6: Representação da consulta Q_2 $((Working >= 5);(Dining);(Home))$ como um *autômato temporizado* no Uppaal.

É interessante observar que a restrição de guarda $y >= 1$ e a reinicialização da variável de *clock*, $y := 0$, aparecem em todas as transições do autômato referente à consulta, forçando o Uppaal a lidar com tempo discreto, implicando que cada episódio da consulta deva durar pelo menos 1 unidade de tempo. A restrição de duração de tempo referente ao predicado *Working* é representada, no autômato, pela restrição de guarda $x >= 5$ na primeira transição, indicando que o objeto móvel deve ter um episódio de pelo menos 5 unidades de tempo no qual é constatada uma atividade de trabalho.

Por último, considere a consulta Q_3, que além de possuir restrição de duração de tempo em um predicado, possui também restrição de duração de tempo entre predicados. O autômato referente à sua consulta é representado na Figura 5.7.

> *Q_3: Quais trajetórias que possuem o seguinte padrão de movimento: realizam uma atividade de trabalho durante pelo menos 2 unidades de tempo, então, depois de no máximo 4 unidades de tempo, estão em casa e, após isto, jantam?*

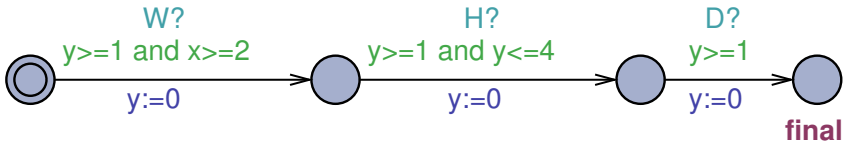


Figura 5.7: Representação da consulta Q_3 $((Working >= 2); [<= 4](Home); (Dining))$ como um *autômato temporizado* no Uppaal.

Na representação da consulta Q_3 (Figura 5.7), são observadas as mesmas restrições que implicam que predicados pertencentes a um determinado episódio da consulta deva durar pelo menos 1 unidade de tempo (i.e. $y >= 1$ e $y := 0$). Para a restrição de duração entre predicados válidos, descrita pelo operador sequencial $;[\sim c]$, pode-se observar a inclusão de outra restrição de guarda - $y <= 4$ - na transição referente ao predicado $Home$ (H). Assim, a restrição referente ao predicado $Home$ só será processada quando o valor da variável de clock y estiver for maior ou igual a 1 e menor ou igual a 4, implicando que o predicado durou pelo menos 1 unidade de tempo e que satisfaz a restrição de duração entre predicados.

Ao final, observa-se que, para cada consulta, tem-se um autômato temporizado específico construído em referência a ela.

5.5 Ambiente de Experimentos

O ambiente de desenvolvimento do protótipo foi constituído da IDE Netbeans 7.2, na qual foi utilizada a linguagem Java para a implementação do protótipo. O kit de desenvolvimento Java utilizado foi o jdk1.6.0_35. O sistema gerenciador de banco de dados utilizado foi o PostgreSQL 8.4.11, que foi executado localmente. A versão da API Java do verificador de modelos Uppaal foi a contida na versão 4.0.13 da ferramenta.

Com relação à configuração do ambiente de testes, estes foram realizados em uma máquina com processador Intel Core 2 Quad de 2.83Ghz x 4, com 3,7 gigabytes de memória principal, executando Ubuntu 12.04 (precise) 64-bit com o kernel linux 3.2.0-31-generic.

5.6 Protótipo

Diante da abordagem de avaliação de consultas sobre trajetórias semânticas, abordada na Seção 4.2, verificou-se a possibilidade do desenvolvimento de um protótipo para a

realização de consultas que utiliza a estratégia de verificação de modelos anteriormente mencionada.

A ideia inicial para o desenvolvimento da interface era possibilitar ao usuário escrever consultas de acordo com a linguagem proposta por este trabalho e executá-las sobre uma base de dados provida de trajetórias semânticas descritas em conformidade com o modelo de dados proposto por este trabalho. Para esta atividade seria necessário à construção de *parser* da nossa linguagem para o modelo lógico descrito pelo verificador de modelo Uppaal. Entretanto, visando à realização de experimentos em um curto espaço de tempo, sem prejuízos para a análise da abordagem proposta, optou-se pela construção de uma interface que possibilitasse o usuário construir as consultas através de um passo-a-passo, facilitando, assim, o processo de tradução para o modelo lógico utilizado pelo Uppaal.

A arquitetura do protótipo apresentado neste trabalho e constituída de 4 módulos, como pode ser analisada na Figura 5.8. A seguir é feita uma explanação sobre cada um dos módulos e suas principais funcionalidades.

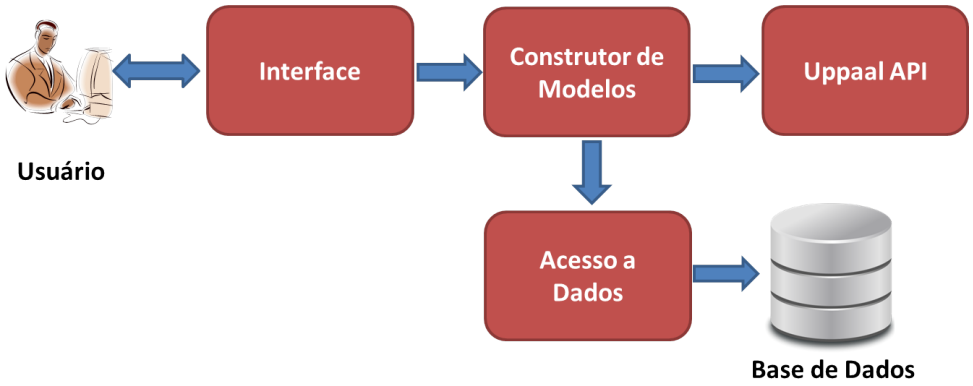


Figura 5.8: Arquitetura do protótipo.

- **Módulo Interface:** é responsável pela apresentação de um formulário de construção da consulta para o usuário. O formulário possibilita que o usuário construa passo-a-passo a consulta expressa de acordo com a linguagem proposta no Capítulo 4. Este módulo utiliza o *framework* Java Swing para a construção das telas e dos componentes gráficos. A comunicação deste módulo com o módulo **Construtor de Modelos** dar-se através da construção dos *autômatos temporizados* referentes a cada trajetória e do *autômato temporizado* referente à consulta construída.

- **Módulo Construtor de Modelos:** é responsável pela criação dos *timed autômatos* referentes a cada trajetória e pela construção do *autômato temporizado* referente à consulta. As trajetórias serão recuperadas de uma base de dados através da comunicação deste módulo com o módulo **Acesso a Dados**, já a construção dos autômatos será realizada através do API do Uppaal. A construção destes autômatos é detalhada na Seção 5.3.

- **Módulo Uppaal API:** este módulo representa a API do Uppaal para a construção dos autômatos, escrita da fórmula lógica e designação do sistema a ser realizada a verificação de modelos.

- **Módulo Acesso a Dados:** é responsável por recuperar os dados da base de dados. Os arquivos de configuração para a indicação da base de dados de trajetórias também é de responsabilidade deste módulo.

O protótipo desenvolvido segue a mesma abordagem exposta na Seção 5.3, entretanto, objetivando a realização dos primeiros testes, optou-se, por questão de limite de tempo, por não abordar a construção da linguagem proposta que inclui restrições temporais entre predicados (i.e. $\langle \text{Moving Pattern} \rangle \rightarrow \langle \text{Episode} \rangle; [\sim c] \langle \text{Moving Pattern} \rangle$), possibilitando que o protótipo seja aprimorado futuramente.

Assim, o protótipo possibilita a realização de consultas de padrão de movimento sobre uma base de dados de trajetórias descritas em conformidade com o modelo de dados do Capítulo 4, utilizando a ideia geral da abordagem proposta na Seção 4.2 e as traduções apresentadas na Seção 5.3. Visando a simplificação da nossa solução utilizando o Uppaal, o Algoritmo 6 apresenta o funcionamento geral do protótipo com relação à avaliação de consultas.

Algoritmo 6: Avaliação da consulta utilizando Verificação de Modelos.

Entrada: Uma consulta q, expressa na nossa linguagem, e um conjunto de trajetórias T.
Saída: Um conjunto R de trajetórias que satisfazem à consulta.

```
1  Construa o autômato Q representativo da consulta q;
2  para t ∈ T faça
3  |  id ⟵ t.id;
4  |  Construa o autômato "Tid" representativo da trajetória t;
5  |  satisfaz ⟵ executaVerificacaoModelo(Q, Tid, E<>Q.final);
6  |  se satisfaz = true então
7  |  |  R ⟵ R ∪ t
8  |  fim se
9  fim para
10 retorna R;
```

O Algoritmo 6 recebe como entrada uma consulta q e um conjunto de trajetórias T. Inicialmente, o algoritmo constrói o autômato representativo da consulta q, chamado de Q, seguindo as regras explicadas na Seção 4.2. Após a construção do autômato de consulta Q, o algoritmo irá construir, para cada trajetória $t \in T$, o seu autômato representativo, que será nomeado como TX, onde X é o id da trajetória. Depois de construído o autômato da trajetória corrente, a verificação de modelos é realizada, através da API do Uppaal, para o sistema constituído do autômato de consulta Q e do autômato da trajetória TX, juntamente com a fórmula $E <> Q.final$, que verifica se o estado $final$ será alcançado no autômato Q. Se o sistema satisfizer a fórmula, então é porque a trajetória satisfaz a consulta e, então, ela será inserida no conjunto resultante, se não, desconsidera a trajetória e verifica a próxima.

5.6.1 Um exemplo de uso da ferramenta

O protótipo construído permite ao usuário realizar a construção da consulta através de um passo-a-passo, não permitindo a construção de consultas não aceitas pela linguagem

proposta neste trabalho.

A construção da consulta baseia-se na definição da linguagem do Capítulo 4, iniciando-se pela seleção de um predicado. Após a seleção do primeiro predicado, outros predicados podem ser selecionados para compôr uma expressão, através dos operadores lógicos. Construída a expressão, esta pode receber uma restrição de duração ou não, constituindo um episódio. Após a construção do episódio, este pode ser adicionado à consulta, construindo um padrão de movimento. A Figura 5.9 apresenta a tela inicial do protótipo, na qual o passo inicial do usuário será escolher um predicado na lista *Predicates* para compor uma expressão. Após isto, o usuário irá acionar o botão *Select*, que irá incluir o predicado no campo de texto denominado de *Predicate*, como apresentado na Figura 5.10.

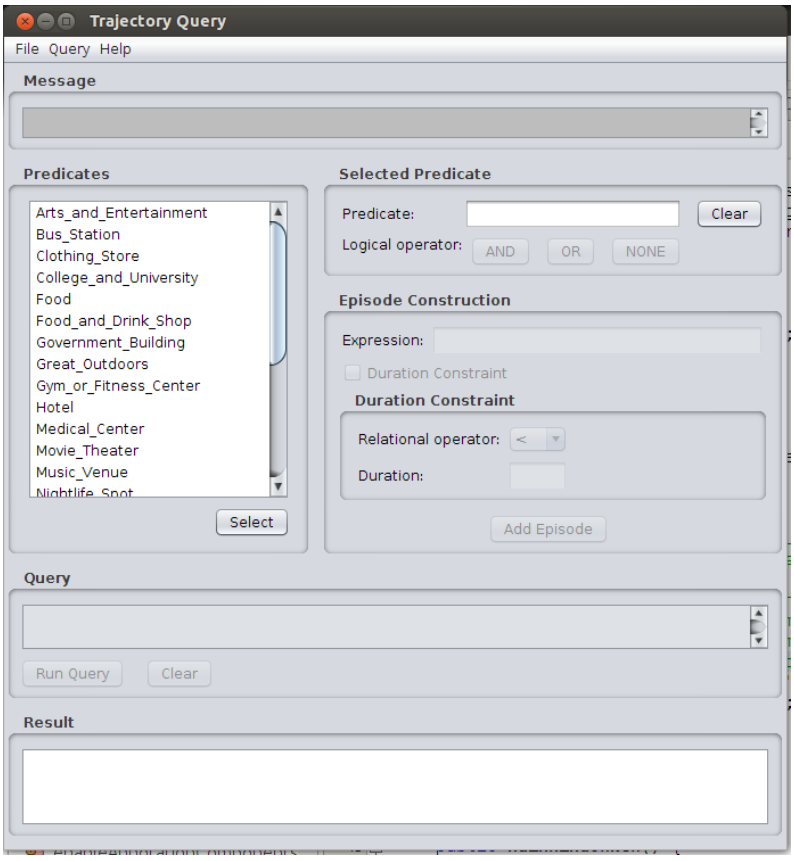


Figura 5.9: Tela inicial do protótipo.

Após a inclusão de um predicado, o usuário terá a opção de adicionar mais predicados para compor a expressão (*Expression*) através dos botões que representam os operadores lógicos: *AND* ou *OR*. Outra opção para o usuário é a não inclusão de mais predicados, que pode ser realizada através do botão *NONE*. A Figura 5.11 apresenta uma expressão composta de operadores lógicos *AND* e *OR*. É possível notar que, após o botão *NONE* ser acionado, os componentes do painel *SelectedPredicate* ficam desabilitados, impossibilitando de o usuário construir uma expressão inválida.

Com uma expressão construída, o usuário pode optar por incluir uma restrição de duração (*DurationConstraint*) ou simplesmente finalizar a construção do episódio, acionando o botão *AddEpisode*. Quando o botão *AddEpisode* é acionado, o protótipo irá verificar se já

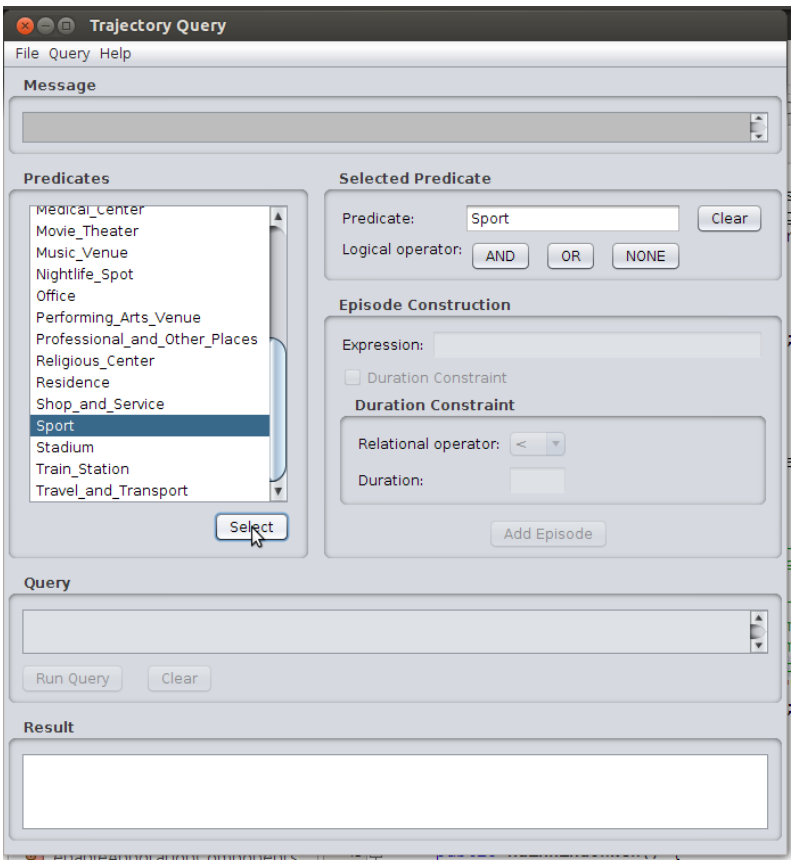


Figura 5.10: Predicado incluído na *textfield Predicate*.

existe um episódio adicionado na consulta, se já houver, o operador sequencial ";" é adicionado antes da adição do próximo episódio, visando à construção de um padrão de movimento. A construção da consulta pode ser observada na Figura 5.12. É importante notar que o operador sequencial $;[\sim c]$ não foi abordado no protótipo, entretanto objetiva-se utilizá-lo em futuras extensões do protótipo.

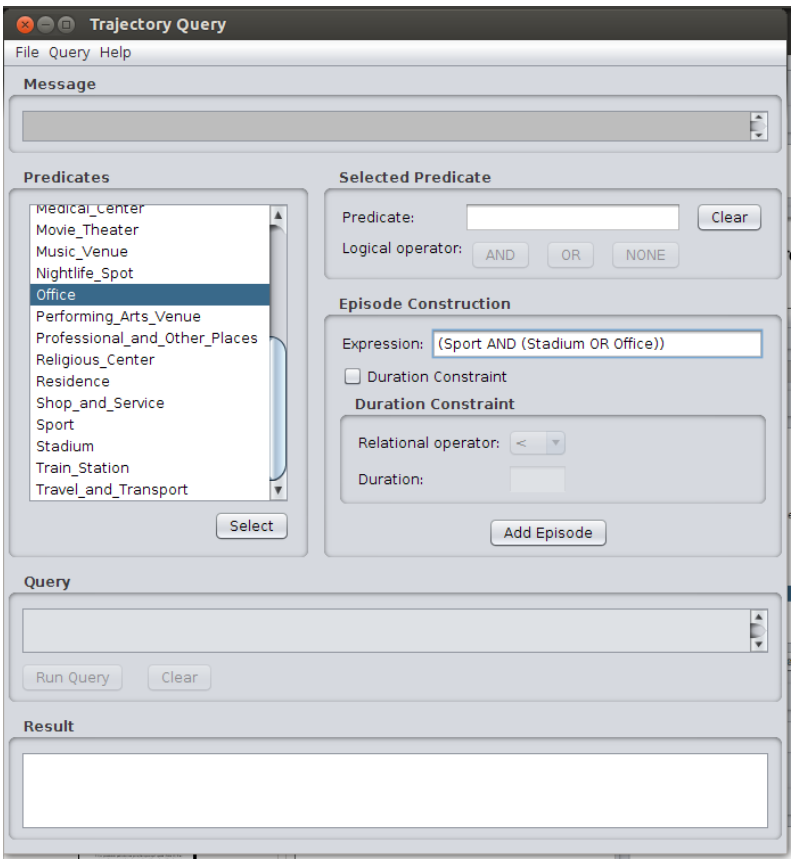


Figura 5.11: Uma expressão da linguagem construída.

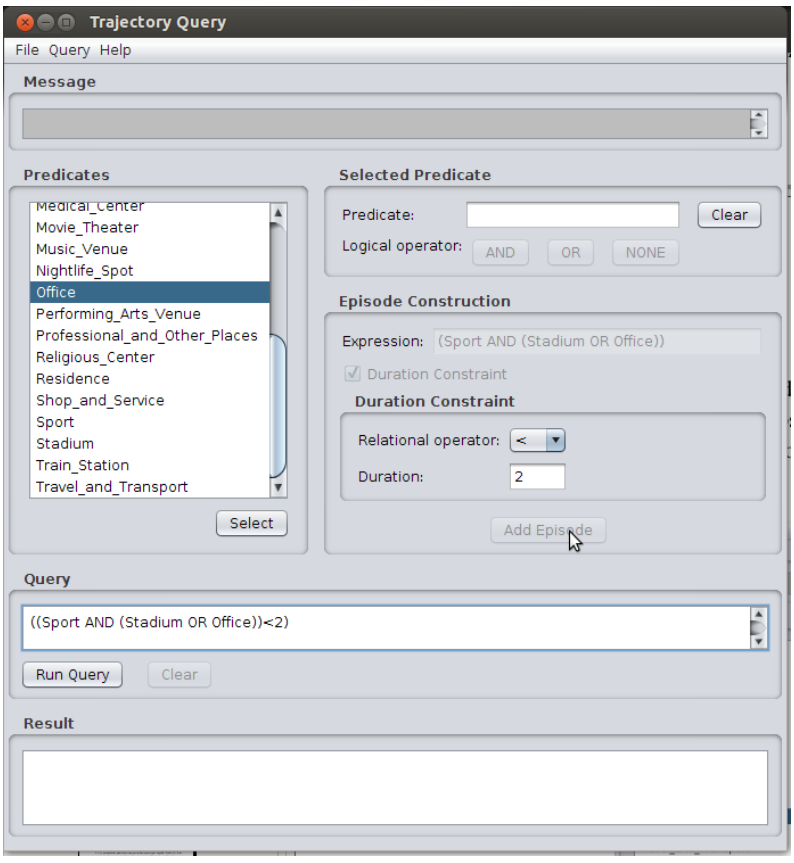


Figura 5.12: Uma consulta construída com duas expressões.

5.7 Dados Utilizados

Para os experimentos, foi utilizado um conjunto de observações posicionais coletadas por uma companhia de seguros italiana, que oferece um desconto aos usuários que possuem um dispositivo GPS embutido em seu carro. O conjunto de dados foi coletado em uma semana em Milão (Itália) e são compostos por 1.806.293 pontos para 17.087 usuários na área da citada cidade. Para os experimentos realizados, os identificadores de usuários serão considerados equivalentes aos identificadores das trajetórias.

Estes dados foram normalizados, visando buscar episódios de **parada** e de **movimento** de cada trajetória. Ao final da normalização, foram encontrados 216.523 episódios de paradas. A partir dos episódios de paradas, foi feita uma análise de quais pontos de interesse do *foursquare*[2] estariam próximos de cada episódio dentro de um raio de 50 metros. Assim, foi criado um evento para cada ponto de interesse pertencente a um episódio de parada. Os eventos pertencentes a uma mesma trajetória e referentes a um mesmo episódio de parada irão compor um episódio da trajetória, como descrito no Capítulo 2.

Ao final, a normalização feita seguiu o esquema apresentado na Figura 5.13. Ou seja, cada trajetória (*semantic_trajectory*) terá como atributos um *id*, o instante de tempo inicial (*startTime*) e o instante de tempo final (*endTime*), além disso uma trajetória tem 1 ou mais eventos (*semantic_trajectory_events*). Os eventos possuem como atributos o id da trajetória (*trajectory_id*), o id do episódio de parada (*episode_id*), o instante inicial do evento (*startTime*), o instante final do evento (*endTime*) e o predicado associado ao evento *predicate*. Para os dados utilizados, o valor do atributo *predicate* será sempre o valor da categoria de um ponto de interesse do *foursquare* associado ao evento. No modelo de dados proposto no Capítulo 2, cada episódio possui um conjunto de predicados associado a ele. No esquema do banco de dados em questão, um episódio $ep = (s, e, \Psi)$ é representado por todas as tuplas da tabela *semantic_trajectory_event* que possuem os mesmos valores para os atributos *semantic_trajectory_id* e *episode_id*. Por exemplo, a trajetória cujo identificador for 1, poderá conter duas tuplas na tabela *semantic_trajectory_event* com o mesmo *episode_id*, mas com predicados diferentes (e.g. Lunch e Home). Cada tupla representaria um predicado associado a um episódio.

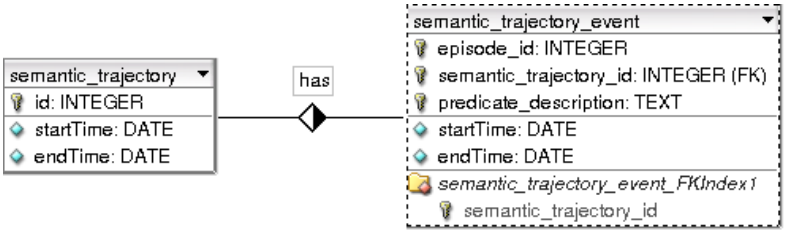


Figura 5.13: Esquema da Base de Dados utilizada nos experimentos.

Ao final da normalização para se obter as trajetórias, conseguimos obter 14.091 trajetórias descritas como uma sequência de episódios. Os episódios distintos, para cada trajetória,

[2] `https://foursquare.com`

somou-se 96.340, enquanto a quantidade total de episódios chegou a 278.346. Estes episódios possuem predicados dos mais diversos, pertencentes a um conjunto de 397 predicados.

A partir destes dados, foi feita uma análise dos dados visando apresentar a distribuição dos dados. Nesta análise foi considerada a quantidade de episódios por trajetória (Tabela 5.1) e a quantidade de predicados, tanto por trajetória, quanto por episódio (Tabela 5.2). É importante ressaltar que uma trajetória pode conter vários episódios e cada episódio pode possuir diversos predicados.

Os valores das tabelas abaixo foram calculados conforme se segue. A média de episódios por trajetória é calculada da seguinte forma: (1) Contamos a quantidade de episódios que cada trajetória possui; (2) Calculamos a média de episódios para todas as trajetórias. A média de predicados por trajetória é calculada da seguinte forma: (1) Contamos a quantidade de predicados que cada episódio distinto possui (episódios distintos são aqueles cujos *semantic_trajectory_id* e *episode_id* são iguais); (2) Computamos a média de predicados para cada trajetória; (3) Computamos a média para todas as trajetórias.

Tabela 5.1: Quantidade de eventos por Trajetórias

Episódios distintos	**Trajetórias**	**Total**
$[50, \infty)$	15	7237
$[40, 50)$	32	
$[30, 40)$	97	
$[20, 30)$	600	
$[10, 20)$	2800	
$[5, 10)$	3693	
$[1, 5)$	6854	6854
Média episódio/traj.	6,84	

Tabela 5.2: Média de predicados por trajetória e por episódio

Média de pred.	**Trajetórias**
$[4, \infty)$	2825
$[3, 4)$	2344
$[2, 3)$	4125
$[1, 2)$	4797
Média pred./episódio	2,8891
Média pred./traj.	2,8664

É interessante observar que, na Tabela 5.1, quase a metade das trajetórias possuem no máximo 5 episódios e que a média de episódios por trajetória é de $6,84$, ou seja, um número considerável para a realização de diversas consultas. Já na Tabela 5.2, é interessante observar que a média de predicados por episódio é de $2,8891$, ou seja, consultas que consideram mais de um predicado por episódio poderão ser realizadas.

5.8 Resultados Obtidos

Os experimentos realizados consistiram-se em buscar os dados de trajetórias da base de dados e depois realizar a consulta, utilizando nossa abordagem (apresentada no Algoritmo 6). Cada consulta foi executada 10 vezes e em seguida foi computada a média de duração da execução destas consultas. Os seguintes casos foram analisados, baseado na metodologia apresentada em 5.1: (i) escalabilidade quanto ao tamanho do banco de dados, (ii) escalabilidade quanto à quantidade de predicados e (iii) escalabilidade quanto à quantidade de predicados com restrição temporal.

Para o primeiro caso (i), buscou-se variar a quantidade de trajetórias a serem trazidas da base de dados para a realização do cálculo de duração. Para este caso utilizou-se uma mesma consulta, que pergunta se existe uma trajetória com o padrão de movimento $(Medical\_Center);(Bus\_Station)$, ou seja, se a trajetória ficou, em algum momento, em um "centro médico" e depois se dirigiu a uma "estação de ônibus". As quantidades de trajetórias utilizadas seguiu-se uma série exponencial na base 10: 10, 100, 1000, 10000, 100000. Para a realização dos testes com 100000 trajetórias, a base de dados apresentada na Seção 5.7 precisou ser replicada. Entretanto, apesar do espaço de memória ter sido expandido na Máquina Virtual Java, os testes não chegaram a serem executados devido a problemas de espaço de memória do Java. Esta falha na execução dos testes com 100000 trajetórias mostra um limitante de nossa abordagem: a memória. Os resultados dos experimentos, incluindo a quantidade de trajetórias, podem ser verificados na Tabela 5.3.

Tabela 5.3: Resultados do processamento de uma consulta utilizando dois predicados, indicando que um Centro Médico foi visitado antes de uma Estação de Ônibus: $((Medical\_Center);(Bus\_Station))$.

Trajetórias	Tempo (nanosegundos)	Tempo (segundos)
10	149294067,20	0.149294067
100	988717568,00	0.988717568
1000	8575179980,80	8.575179981
10000	81528029184,00	81.52802918

Os resultados dos experimentos realizados para o primeiro caso, apresentados na Tabela 5.3, apresentam os tempos médios para 10 execuções da abordagem, considerando a quantidade de trajetórias. Visando facilitar a analise da evolução do tempo em decorrência do tamanho da base de dados, foi construído o gráfico apresentado na Figura 5.14 baseado na Tabela 5.3. Este gráfico utilizou a escala logarítmica na base 10 para facilitar a identificação do tipo de crescimento do gráfico.

Observa-se no gráfico que o tempo aumenta na mesma proporção que aumenta a quantidade de trajetórias, ou seja, o crescimento do tempo gasto é linear.

Para o segundo caso (ii), buscou-se variar a quantidade de predicados na consulta para uma quantidade fixa de trajetórias. Como primeiro experimento para este caso, considere a quantidade de trajetórias igual a 10 e as quantidades de predicados utilizadas iguais aos valores de uma série exponencial: 2, 4, 8, 16 e 32.

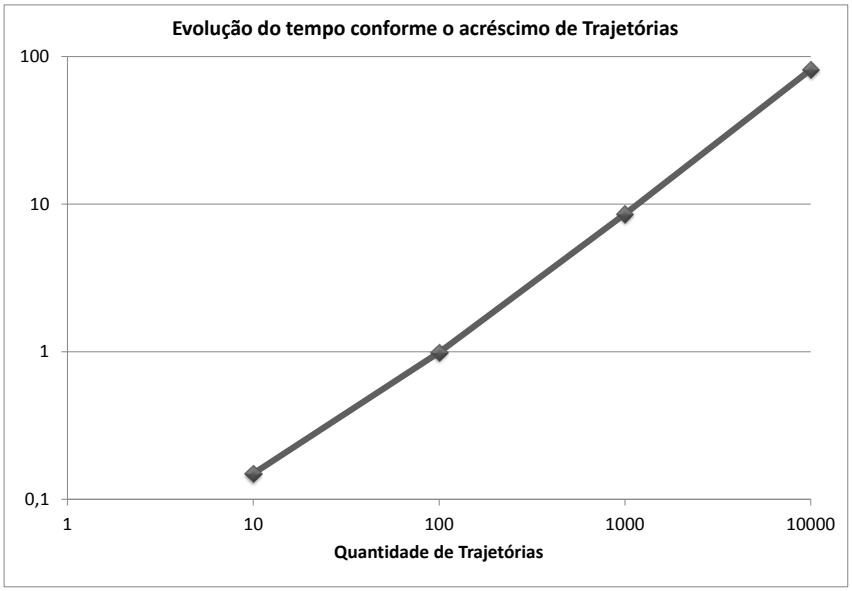


Figura 5.14: Tempo em decorrência da quantidade de trajetórias.

As consultas apresentadas abaixo foram utilizadas no experimento que se segue. Note que cada consulta apresenta uma quantidade distinta de predicados.

- **2 predicados:** $(Medical\_Center);(Bus\_Station)$
- **4 predicados:** $(Medical\_Center);(Bus\_Station);(Shop\_and\_Service);$ $(Travel\_and\_Transport)$
- **8 predicados:** $(Medical\_Center);(Bus\_Station);(Shop\_and\_Service);$ $(Travel\_and\_Transport);(College\_and\_University);(Food);(Clothing\_Store);$ $(Professional\_and\_Other\_Places)$
- **16 predicados:** $(Medical\_Center);(Bus\_Station);(Shop\_and\_Service);$ $(Travel\_and\_Transport);(College\_and\_University);(Food);(Clothing\_Store);$ $(Professional\_and\_Other\_Places);(Nightlife\_Spot);(Food);(Bus\_Station);$ $(Shop\_and\_Service);(College\_and\_University);(Travel\_and\_Transport);$ $(Shop\_and\_Service);(Food)$
- **32 predicados:** $(Medical\_Center);(Bus\_Station);(Shop\_and\_Service);$ $(Travel\_and\_Transport);(College\_and\_University);(Food);(Clothing\_Store);$ $(Professional\_and\_Other\_Places);(Nightlife\_Spot);(Food);(Bus\_Station);$ $(Shop\_and\_Service);(College\_and\_University);(Travel\_and\_Transport);$ $(Shop\_and\_Service);(Food);(Medical\_Center);(Bus\_Station);(Shop\_and\_Service);$ $(Travel\_and\_Transport);(College\_and\_University);(Food);(Clothing\_Store);$ $(Professional\_and\_Other\_Places);(Nightlife\_Spot);(Food);(Bus\_Station);$ $(Shop\_and\_Service);(College\_and\_University);(Travel\_and\_Transport);$ $(Shop\_and\_Service);(Food)$

Os resultados dos experimentos, incluindo a quantidade variada de predicados, podem ser observados na Tabela 5.4.

Tabela 5.4: Resultados variando a quantidade de predicados para 10 trajetórias.

Predicados	**Tempo (segundos)**
2	0.149294067
4	0.153450726
8	0.147967885
16	0.16282135
32	0.203121677

Para um segundo cenário do caso (ii), agora considerando uma quantidade fixa de trajetórias igual a 100, os valores do tempo transcorrido foram iguais aos apresentados na Tabela 5.5.

Tabela 5.5: Resultados variando a quantidade de predicados para 100 trajetórias.

Predicados	**Tempo (segundos)**
2	0.988717568
4	1.011174093
8	1.082930176
16	1.225154458
32	1.51121664

Em um terceiro cenário do caso (ii), a quantidade de trajetórias foi fixada em 1000 e a quantidade de predicados foi variando, de forma semelhante aos cenários anteriores. Os valores dos tempos transcorridos podem ser verificados na Tabela 5.6.

Tabela 5.6: Resultados variando a quantidade de predicados para 1000 trajetórias.

Predicados	**Tempo (segundos)**
2	8.575179981
4	8.992402637
8	9.75721513
16	11.07160924
32	13.73446636

Finalmente, em um último cenário analisado para o caso (ii), foi fixada a quantidade de trajetórias a serem buscadas em 10000. Os predicados foram variando em quantidades, conforme os casos anteriores. Os valores dos tempos transcorridos para este cenário podem ser verificados na Tabela 5.7

Todos os resultados dos experimentos referentes aos cenários do caso (ii) são verificados na Figura 5.15. Pode-se observar que a taxa de crescimento do tempo de execução da abordagem permanece linear.

Finalmente, para o terceiro caso (iii), foram realizadas consultas com 32 predicados em uma base de dados constituída de 10000 (dez mil) trajetórias semânticas. A primeira

Tabela 5.7: Resultados variando a quantidade de predicados para 10000 trajetórias.

Predicados	**Tempo (segundos)**
2	81.52802918
4	84.05521531
8	91.34277263
16	105.1521647
32	133.347292

consulta não considera nenhum predicado com restrição temporal, que no nosso caso será uma restrição do tipo "pelo menos 2 unidades de tempo" (ou seja, $>= c$, onde $c \in \mathbb{N}$). A segunda consulta terá restrições temporais nos dois primeiros predicados e as consultas posteriores terão a quantidade de predicados com restrições temporais iguais aos valores restantes da sequência exponencial de 2 utilizada: 4, 8, 16 e 32. Os valores da quantidade de tempo despendida em cada caso são listados na Tabela 5.8.

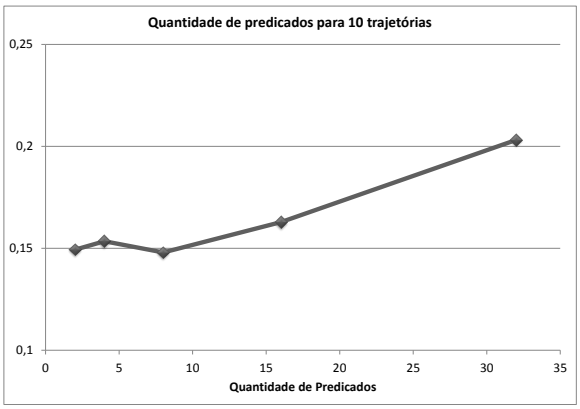
Tabela 5.8: Resultados para um padrão de movimento constituído de 32 predicados a ser buscado em uma base de dados de 10000. A quantidade de predicados com restrições temporais é incrementada a cada nova execução

Predicados com restrições temporais	**Tempo (segundos)**
0	133.347292
2	133.7232065
4	135.0178832
8	136.4382188
16	140.1749111
32	146.623837

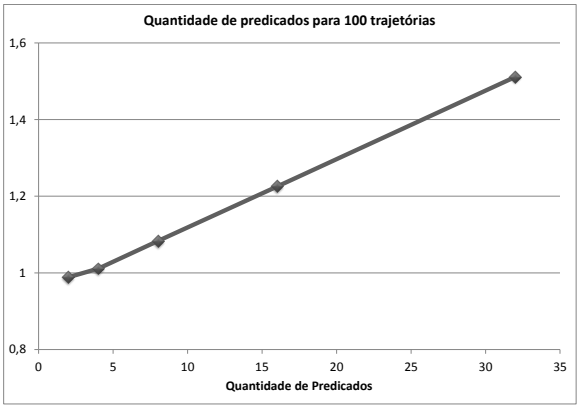
Observa-se no gráfico da Figura 5.16 que o tempo aumenta na mesma proporção do acréscimo do número de predicados com restrições temporais presentes na consulta, ou seja, o crescimento do tempo gasto é linear.

5.9 Conclusão

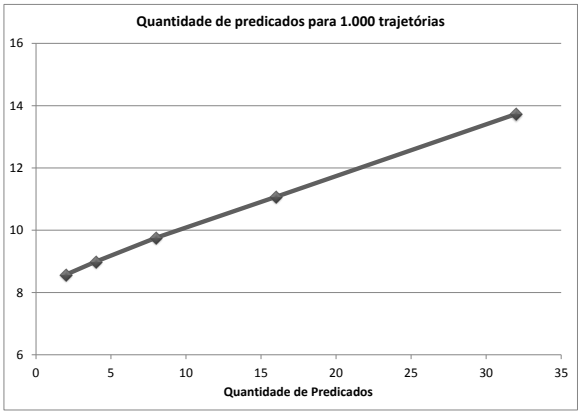
Neste capítulo foi apresentada uma implementação da solução proposta no Capítulo 4, a qual se utilizou do verificador de modelos Uppaal. Inicialmente, a definição dos experimentos a serem realizados é apresentada na Seção 5.1. Em seguida, na Seção 5.2, é justificada a escolha do verificador de modelos Uppaal. A seguir, é apresentada uma tradução da solução para o verificador de modelos, enfatizando que a informação sobre os instantes, ao longo do tempo, no qual um determinado episódio ocorreu (e.g. "Correndo" às $3hs$), não se conseguiu expressar. Na Seção 5.5 é esclarecido onde foram realizados os experimentos. Posteriormente, informações sobre a implementação do protótipo, assim como um exemplo do uso da ferramenta são apresentados, respectivamente, nas Seções 5.6 e 5.6.1. Após isto, informações sobre os dados reais utilizados nos experimentos são expostas na Seção 5.7. Finalmente, os resulta-



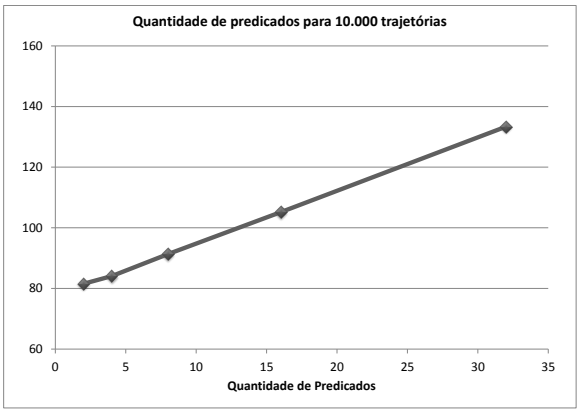
(a) 10 Trajetórias



(b) 100 Trajetórias



(c) 1000 Trajetórias



(d) 10000 Trajetórias

Figura 5.15: Tempo em decorrência da quantidade de predicados para cada quantidade de trajetórias semânticas no banco.

dos dos experimentos são apresentados na Seção 5.8, mostrando que a abordagem é linear no tamanho do banco de dados e na quantidade de predicados presentes na consulta, muito embora a existam muitas trajetórias com poucos episódios e poucas trajetórias com muitos episódios, constituindo em uma não homogeneidade dos dados. Conclui-se que base de dados mais homogêneas, nas quais a maioria das trajetórias possuam muitos episódios, são necessárias para conclusões mais precisas acerca dos resultados.

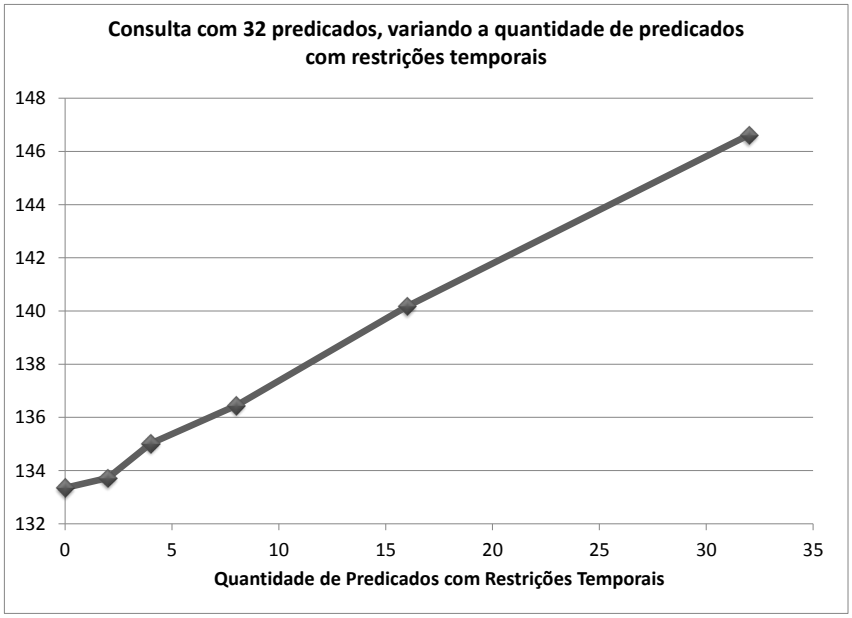


Figura 5.16: Tempo em decorrência da quantidade de predicados com restrição temporal para 10000 (dez mil) trajetórias semânticas.

6 CONCLUSÃO E TRABALHOS FUTUROS

6.1 Resumo dos Capítulos

Nesta dissertação foi apresentado um estudo sobre o conceito de trajetória semântica introduzido por (SPACCAPIETRA et al., 2008). A partir deste estudo verificou-se que diversos trabalhos propõem soluções para o enriquecimento semântico de trajetórias brutas, transformando-as em trajetórias semânticas descritas por uma sequência de eventos. Entretanto, apesar de ser possível obter trajetórias semânticas a partir de métodos propostos por estes trabalhos, a utilidade de dados de trajetórias semanticamente anotados não é enfatizada, principalmente em como realizar consultas sobre estes dados. Particularmente, o trabalho focou-se em consultas que pudessem expressar e recuperar trajetórias que atendessem a determinados padrões de movimento. Um padrão de movimento foi definido, neste trabalho, como uma sequência de predicados que podem ser válidos durante um intervalo de tempo.

Neste contexto, este trabalho definiu, primeiramente, o **problema de expressar consultas sobre padrões de movimento e processar tais consultas no sentido de recuperar trajetórias que correspondam a tais padrões** no Capítulo 2, apresentando a fundamentação teórica necessária para entender o problema. Em seguida, foram apresentados e discutidos os principais trabalhos relacionados com o problema exposto neste trabalho, no Capítulo 3. As principais contribuições desta dissertação foram apresentadas no Capítulo 4, onde foram propostas uma abordagem para o processamento de consultas sobre trajetórias semânticas baseada em lógica temporal e uma linguagem de consulta. Toda a contribuição exposta no Capítulo 4 foi baseada no modelo de dados definido no Capítulo 2, no qual a trajetória é definida como uma sequência de episódios nos quais um conjunto de predicados é válido. Para a validação do trabalho, foi implementado um protótipo para a realização de experimentos, que foram apresentados no Capítulo 5. Resultados preliminares, baseados nos experimentos realizados com dados reais, mostraram que a abordagem é tão eficiente, do ponto de vista computacional, quanto outros trabalhos.

6.2 Objetivos Alcançados

No início desta dissertação, objetivos a serem alcançados foram listados, visando resolver o problema de verificar quais as trajetórias semânticas, em um banco de dados, correspondem a um dado padrão de movimento de entrada. Durante todo o trabalho realizado, procurou-se alcançar todos os objetivos específicos, objetivando solucionar o problema. Abaixo, cada objetivo é listado e uma descrição do que foi feito para atingi-lo é apresentada:

- Definição de um modelo de dados que permita representar as trajetórias semânticas em um banco de dados: este objetivo foi alcançado através de um modelo de dados proposto no Capítulo 2, o qual descreve uma trajetória semântica como uma sequência de episódios compreendidos por instantes de tempo inicial e final, onde um conjunto de predicados são válidos;

- Definição de uma abordagem para o processamento das consultas de padrões de movimento: este objetivo foi alcançado através da solução baseada em verificação de modelos para a lógica TCTL, proposta na Seção 4.2 do Capítulo 4. Este foi um dos objetivos mais difíceis de ser atingido, pois quando se optou pelo uso de lógica temporal para a definição do padrão de movimento, diversos problemas relacionados à compatibilidade entre a expressividade da lógica e do padrão de movimento definido foram encontrados. Outro desafio foi com relação à complexidade de se processar a consulta de forma eficiente e que não resultasse na redução de expressividade da mesma, o que acabou ocorrendo. O problema de expressividade encontrado refere-se à impossibilidade de se trabalhar com diversos níveis de granularidade de tempo (e.g. segundos, minutos, horas) conjuntamente;

- Especificação e descrição da semântica da linguagem de consultas para padrões de movimento sobre trajetórias semânticas: este objetivo foi alcançado pela especificação da linguagem apresentada na Seção 4.3 do Capítulo 4. A linguagem proposta é classificada como um *syntactic sugar*, simplificando as fórmulas TCTL utilizadas para descrever padrões de movimento na abordagem proposta. Aqui se verificou que o grande desafio foi adequar a especificação da linguagem com a solução proposta para o seu processamento;

- Analisar a eficiência da abordagem do ponto de vista computacional e compará-la com outras abordagens: este objetivo é atingido através da Seção 4.2.1, na qual a complexidade da abordagem é analisada, e através da construção de um protótipo que visa à concretização real da proposta para a realização de experimentos com dados reais (Capítulo 5). O protótipo utilizou a API do verificador de modelos Uppaal para possibilitar a verificação de modelos realizada pela solução proposta neste trabalho. Na análise dos resultados, percebeu-se que a memória foi um fator limitante, já que testes para banco de dados com 100.000 trajetórias semânticas não puderam ser realizados. Ao final, mostrou-se que a abordagem proposta pode ser executada em tempo polinomial e que, do ponto de vista prático, através de experimentos, os tempos de consultas crescem de forma linear para um conjunto de dados nos quais as trajetórias possuem poucos episódios, em média $6,84$ episódios. Resultados de melhores qualidades poderão ser obtidos por conjuntos de dados maiores, nos quais as trajetórias sejam mais complexas, possuindo um número maior de episódios e predicados.

6.3 Comparação com outros trabalhos

Após as análises realizadas através de experimentos e do formalismo da abordagem de processamento de consultas abordada neste trabalho, verificou-se que a solução proposta foi a única a abordar trajetórias semânticas como objetos de primeira classe e um processamento de consultas baseado em verificação de modelos. Neste contexto, espera-se que este trabalho seja um motivador de pesquisas na linha de estudos voltada para a representação lógica de linguagens de consulta sobre trajetórias, assim como na linha de estudos voltada para modelos formais de verificação.

Objetivando comparar o trabalho desenvolvido por esta dissertação com os traba-

lhos relacionados, a Tabela 6.1 foi construída. Esta tabela é semelhante à apresentada no Capítulo 3, entretanto uma coluna adicional, com a proposta desta dissertação, foi adicionada.

6.4 Trabalhos Futuros

Após a análise dos objetivos alcançados, verificaram-se inúmeras oportunidades de trabalhos futuros. A primeira destas oportunidades refere-se ao incremento da solução proposta na Seção 4.2 do Capítulo 4, visando possibilitar outras restrições temporais, como a restrição de que um predicado ocorra em um dado instante de tempo (e.g. "Correndo" às 3*hs*) e a verificação se um determinado predicado ocorre no instante de tempo imediatamente após a ocorrência de outro predicado.

Trabalhos futuros também foram verificados na construção do protótipo e com relação aos experimentos, abordados no Capítulo 5. Na construção do protótipo se optou, por questão de limite de tempo, por não incluir, na construção da linguagem proposta, restrições temporais entre predicados (i.e. $\langle$Moving Pattern$\rangle \rightarrow \langle$Event$\rangle;[\sim c]\langle$Moving Pattern$\rangle$). Com relação aos experimentos, espera-se resolver o problema de limitação de memória para grandes bases de dados, como, por exemplo, 100.000 trajetórias semânticas. Possíveis soluções para este problema poderiam incluir a utilização de índices e o desenvolvimento de algoritmos que possibilitassem utilizar o disco rígido. A respeito da utilização de índices, estes seriam utilizados para a realização de *pruning* nos dados armazenados no banco, antes de utilizar a abordagem proposta para processamento de consultas. Os índices auxiliariam a filtrar somente as trajetórias semânticas que contivessem predicados presentes no padrão de movimento a ser buscado. Esta estratégia poderia favorecer a obtenção de resultados melhores em futuros testes. Ainda com relação aos experimentos, a utilização de dados nos quais poucas trajetórias possuem muitos episódios distintos e muitas trajetórias possuem poucos episódios, podem ter influenciado a linearidade dos resultados, indicando que uma distribuição mais homogênea de episódios por trajetórias podem fazer com que os experimentos apresentem resultados de melhores qualidades.

Outro trabalho futuro relacionado com os experimentos é a realização de testes que envolvam restrições temporais e operadores lógicos entre os predicados, além de implementar outros algoritmos, como o KMP (MOUZA; RIGAUX; SCHOLL, 2005) (utilizado em correspondência de padrões), a fim de realizar comparações.

Outra oportunidade de trabalho futuro seria a definição de uma álgebra para consultas sobre trajetórias semânticas.

Conclui-se, então, que o presente trabalho propôs uma nova abordagem para a realização de consultas sobre dados de trajetórias semânticas, visando buscar as trajetórias que correspondam a um determinado padrão de movimento e que diversas oportunidades de estudos futuros, visando à realização destas consultas puderam ser observadas.

Tabela 6.1: Analise comparativa entre os trabalhos relacionados e a proposta desta dissertação.

Características	**Dindar**	**Cadonna**	**Vieira**	**Guting**	**Vaisman**	**Proposta da Dissertação**
Predicados com restrições temporais	Sim	Não	Restrição de intervalo onde ocorreu determinado predicado.	Restrição de duração e de instante de tempo em predicados e entre eles	Restrições temporais nos valores de atributos de tempo	Restrição de duração em predicados e entre eles.
Solução voltada para Trajetórias	Não	Não	Trajetórias Brutas	Moving Point (função que dado um intervalo de tempo, obtêm-se a representação geométrica da trajetória)	Brutas, com avaliação sobre uma representação semântica	Trajetórias semânticas como objeto de primeira classe
Tipo de processamento de consultas	Baseada em autômato	Baseada em autômato	Baseada em estrutura de índices	Baseado na solução do problema da satisfação de restrições	Baseada em autômato	Baseado em verificação de modelos de autômatos temporizados
Complexidade do processamento	Não se sabe	Exponencial	Polinomial	Exponencial: sem usar índices; Polinomial: com índices (prova feita através de experimentos)	Exponencial	Polinomial
A consulta permite a verificação de mais de um predicado por intervalo de tempo	Não	Não	Não	Sim	Sim	Sim
Principal desvantagem	Verificar a ocorrência de mais de um predicado por intervalo de tempo	Processamento de consultas	Não lida com restrições de duração	Experimentos com poucos predicados dependentes do tempo	Processamento de consultas	Memória

REFERÊNCIAS BIBLIOGRÁFICAS

ALUR, R.; DILL, D. Automata for modeling real-time systems. *Automata, languages and programming*, Springer, p. 322–335, 1990.

ALUR, R.; DILL, D. A theory of timed automata. *Theoretical computer science*, Elsevier, v. 126, n. 2, p. 183–235, 1994.

ALVARES, L.; BOGORNY, V.; KUIJPERS, B.; MACELO, J. de; MOELANS, B.; PALMA, A. Towards semantic trajectory knowledge discovery. *Relatório técnico. Hasselt University*, Citeseer, p. 12, 2007.

ALVARES, L. O.; BOGORNY, V.; KUIJPERS, B.; MACEDO, J. A. F. de; MOELANS, B.; VAISMAN, A. A model for enriching trajectories with semantic geographical information. *ACM GIS '07*, ACM Press, New York, New York, USA, p. 1, 2007. Disponível em: <http://portal.acm.org/citation.cfm?doid=1341012.1341041>.

BAGLIONI, M.; MACEDO, J.; RENSO, C.; WACHOWICZ, M. An ontology-based approach for the semantic modelling and reasoning on trajectories. *Advances in Conceptual Modeling–Challenges and Opportunities*, Springer, p. 344–353, 2008.

BAIER, C.; KATOEN, J. et al. *Principles of model checking*. [S.l.]: MIT press, 2008.

BEHRMANN, G.; DAVID, A.; LARSEN, K. *A Tutorial on Uppaal 4.0*. 2006. Disponível em: <http://www.uppaal.org>.

BELLINI, P.; MATTOLINI, R.; NESI, P. Temporal logics for real-time system specification. *ACM Computing Surveys (CSUR)*, ACM, v. 32, n. 1, p. 12–42, 2000.

BENGTSSON, J.; YI, W. Timed automata: Semantics, algorithms and tools. *Lectures on Concurrency and Petri Nets*, Springer, p. 87–124, 2004.

BOGORNY, V.; KUIJPERS, B.; ALVARES, L. St-dmql: A semantic trajectory data mining query language. *International Journal of Geographical Information Science*, Taylor & Francis, v. 23, n. 10, p. 1245–1276, 2009.

BOUYER, P.; LAROUSSINIE, F. Model checking timed automata. *Modeling and Verification of Real-Time Systems*, Wiley Online Library, p. 111–140, 2010.

CADONNA, B.; GAMPER, J.; BÖHLEN, M. H. Sequenced Event Set Pattern Matching Categories and Subject Descriptors. In: *Proceedings of the 14th EDBT*. [S.l.: s.n.], 2011. p. 33—-44.

CAO, X.; CONG, G.; JENSEN, C. Mining significant semantic locations from gps data. *Proceedings of the VLDB Endowment*, VLDB Endowment, v. 3, n. 1-2, p. 1009–1020, 2010.

CIMATTI, A.; CLARKE, E.; GIUNCHIGLIA, F.; ROVERI, M. Nusmv: A new symbolic model verifier. In: SPRINGER. *Computer Aided Verification*. [S.l.], 1999. p. 682–682.

DINDAR, N. *Pattern Matching over Sequences of Rows in a Relational Database System*. 53 p. Dissertação (Mestrado) — ETH Zurich, Switzerland, 2008.

EBBINGHAUS, H.; FLUM, J.; THOMAS, W. *Mathematical logic*. [S.l.]: Springer, 1994.

EMERSON, E. Temporal and modal logic. *Handbook of theoretical computer science*, Amsterdam, v. 2, p. 995–1072, 1990.

GARSON, J. *Modal Logic*. 2009. `http://plato.stanford.edu/entries/logic-modal/`. [Online; acessado em 17-Novembro-2012].

GOMEZ, L. I.; VAISMAN, A. a. Efficient constraint evaluation in categorical sequential pattern mining for trajectory databases. In: *12th EDBT '09*. New York, NY, USA: ACM Press, 2009. p. 541–552. ISBN 9781605584225. Disponível em: <http://portal.acm.org/citation.cfm?doid=1516360.1516423>.

GÜTING, R.; BÖHLEN, M.; ERWIG, M.; JENSEN, C.; LORENTZOS, N.; SCHNEIDER, M.; VAZIRGIANNIS, M. A foundation for representing and querying moving objects. *ACM Transactions on Database Systems (TODS)*, ACM, v. 25, n. 1, p. 1–42, 2000.

GÜTING, R.; SCHNEIDER, M. *Moving objects databases*. [S.l.]: Morgan Kaufmann Pub, 2005. 258 p. ISBN 0120887991.

HUTH, M.; RYAN, M. *Logic in Computer Science: Modelling and reasoning about systems*. [S.l.]: Cambridge Univ Pr, 2004.

LAROUSSINIE, F.; MARKEY, N.; SCHNOEBELEN, P. Model checking timed automata with one or two clocks. *CONCUR 2004-Concurrency Theory*, Springer, p. 387–401, 2004.

LARSEN, K.; PETTERSSON, P.; YI, W. Uppaal in a nutshell. *International Journal on Software Tools for Technology Transfer (STTT)*, Springer, v. 1, n. 1, p. 134–152, 1997.

MARKEY, N.; SCHNOEBELEN, P. Tsmv: A symbolic model checker for quantitative analysis of systems. In: IEEE. *Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings. First International Conference on the*. [S.l.], 2004. p. 330–331.

MEHLER, T. *Challenges and Applications of Assembly-Level Software Model Checking*. 173 p. Dissertação (Mestrado) — Universität Dortmund, Germany, 2005.

MOUZA, C. du; RIGAUX, P.; SCHOLL, M. Efficient evaluation of parameterized pattern queries. In: ACM. *Proceedings of the 14th ACM international conference on Information and knowledge management*. [S.l.], 2005. p. 728–735.

PELEKIS, N.; THEODORIDIS, Y.; VOSINAKIS, S.; PANAYIOTOPOULOS, T. Hermes–a framework for location-based data management. *Advances in Database Technology-EDBT 2006*, Springer, p. 1130–1134, 2006.

SAKR, M.; GÜTING, R. Spatiotemporal pattern queries. *GeoInformatica, online first*, Springer, 2010.

SIMÕES, D.; MACEDO, J. D. Uma linguagem de consulta para padrões espaço-temporais em trajetórias semânticas. *Proceedings of 26th Brazilian Symposium on Databases - Short Paper*, Sociedade Brasileira de Computação, 2011.

SIMÕES, D.; VIANA, H.; MARKEY, N.; MACEDO, J. D. Querying trajectories through model checking based on timed automata. *Proceedings of 27th Brazilian Symposium on Databases - Short Paper*, Sociedade Brasileira de Computação, p. 33–40, 2012.

SPACCAPIETRA, S.; PARENT, C.; DAMIANI, M.; MACEDO, J. D.; PORTO, F.; VANGENOT, C. A conceptual view on trajectories. *Data & knowledge engineering*, Elsevier, v. 65, n. 1, p. 126–146, 2008.

VIEIRA, M. R.; BAKALOV, P.; TSOTRAS, V. J. Querying Trajectories Using Flexible Patterns. In: *Proceedings of the 13th EDBT*. [S.l.: s.n.], 2010. p. 406–417.

XIE, K.; DENG, K.; ZHOU, X. From trajectories to activities: a spatio-temporal join approach. In: ACM. *Proceedings of the 2009 International Workshop on Location Based Social Networks*. [S.l.], 2009. p. 25–32.

YAN, Z.; CHAKRABORTY, D.; PARENT, C.; SPACCAPIETRA, S. SeMiTri: A Framework for Semantic Annotation of Heterogeneous Trajectories. In: *Proceedings of the 14th EDBT*. [s.n.], 2011. Disponível em: <http://infoscience.epfl.ch/record/163851/files/201103_-EDBT.pdf>.

YAN, Z.; PARENT, C.; SPACCAPIETRA, S.; CHAKRABORTY, D. A Hybrid Model and Computing Platform for Spatio-semantic Trajectories. *The Semantic Web: Research and Applications*, p. 60–75, 2010.

ZEMKE, F.; WITKOWSKI, A.; CHERNIAK, M.; COLBY, L. *Pattern matching in sequences of rows*. [S.l.], 2007. 1–29 p.