On Qualitative Route Descriptions: Representation and Computational Complexity

Matthias Westphal,
Stefan Wölfl and **Bernhard Nebel**
Department of Computer Science
University of Freiburg, Germany
{westpham,woelfl,nebel}@informatik.uni-freiburg.de

Jochen Renz
Research School of Computer Science
The Australian National University
Canberra, Australia
jochen.renz@anu.edu.au

Abstract

The generation of route descriptions is a fundamental task of navigation systems. A particular problem in this context is to identify routes that can easily be described and processed by users. In this work, we present a framework for representing route networks with the qualitative information necessary to evaluate and optimize route descriptions with regard to ambiguities in them. We identify different agent models that differ in how agents are assumed to process route descriptions while navigating through route networks. Further, we analyze the computational complexity of matching route descriptions and paths in route networks in dependency of the agent model. Finally we empirically evaluate the influence of the agent model on the optimization and the processing of route instructions.

1 Introduction

One of the most widely used applications of spatial data in everyday life is route navigation. Since GPS receivers have been integrated into many devices, spatial data about routes or spatial traces of events have become widely available. At the same time, there is a growing interest in such data for applications ranging from constructing map information (e.g., OpenStreetMap[1]) to sharing GPS traces of biking or hiking tours (e.g., EveryTrail[2]). While many of these applications require metric data to be present at runtime, the evaluation and generation of *qualitative descriptions* of routes can be performed on a qualitative representation of preprocessed metric data.

In the literature several approaches to representing spatial route information qualitatively have been discussed. For example, Kuipers' Spatial Semantic Hierarchy [2000] provides a model for the representation of spatial information that integrates qualitative and quantitative levels of description. Based on ideas of the Spatial Semantic Hierarchy, Krieg-Brückner *et al.* [2004] introduce a concept of route graph that is tailored for the semantic representation of route instructions in the context of human-robot interaction.

[1] http://www.openstreetmap.org/
[2] http://www.everytrail.com/

Contrary to such high-level representations of spatial route information the literature on route descriptions is often based on rather low-level graph-theoretical notions. The focus is usually on criteria to evaluate the quality of route descriptions, such as route length, simplicity, description length, and reliability. The starting point in this research direction is the work by Mark [1986]. He proposes to utilize A^* with a weighted sum of metric length and path complexity. The complexity of a path is here estimated by a sum of penalty values assigned to the intersections on a path according to a "frame and slot" representation. Following this idea, Duckham and Kulik [2003] minimize the complexity of a route description by a shortest path algorithm, where the cost function accounts for the amount of information required to negotiate each street intersection. Several different (cognitive) cost functions can be used in this context.

Since qualitative route descriptions are often ambiguous, i.e., an agent processing a description may face situations, where several options are consistent with the description, it seems natural to consider cost functions that take ambiguities into account (see, e.g., [Haque *et al.*, 2006]). Further aspects discussed in the literature include the role of landmarks in route descriptions (e.g., [Duckham *et al.*, 2010]) as well as strategies to generate compact route description by spatial chunking [Richter and Duckham, 2008].

While most work on route descriptions is motivated by cognitive aspects of human wayfinding, the focus of this paper is on the computational aspects that arise when route descriptions and paths in the underlying route network are to be matched. We propose to investigate such matching problems by an approach that studies different agent models and hence different execution strategies within a simple graph-like representation of the route network. Our approach takes into account the ambiguities when generating or evaluating route descriptions. This paves the way for many interesting evaluation and optimization criteria, such as "What are the chances of reaching the desired destination given a description?" or "What is the *expected* distance to the destination after following a description?"

The contribution of this paper is twofold. Firstly, we introduce a concept of decision frame that is adapted from a more general notion of route graph presented in [Renz and Wölfl, 2010]. Decision frames are tailored towards a compact representation of qualitative route information that al-

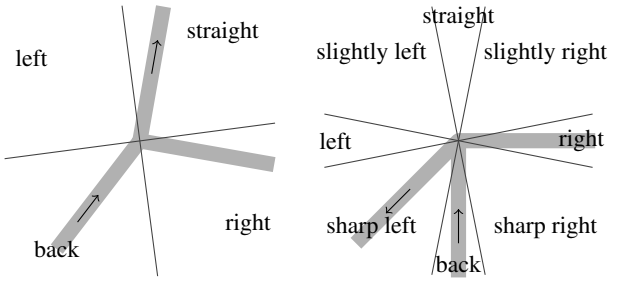

Figure 1: Two different qualitative schemata used to define turn actions at a street intersection.

lows for evaluating and optimizing qualitative route descriptions. Secondly, we focus on ambiguities in route descriptions, i.e., descriptions that leave several alternatives to the agents processing them. To this end, we discuss properties of agents processing route descriptions that influence the potential paths taken in a route network. We distinguish different types of agents and analyze the computational complexity for generating and for processing route descriptions with respect to these agent types. Finally, we present empirical results that demonstrate the practical differences between the agent types.

In the following section we discuss how a qualitative representation of an agent's decisions can be extracted from the layout of a route network. Then, in Section 3 we present different agent models for interpreting route descriptions. In Section 4, the computational complexity of evaluating and optimizing route descriptions, in dependency of the agent model, is discussed. We report on first empirical results on the effects of the agent model in Section 5. Section 6 summarizes our results.

2 Representation and Decision Frames

2.1 Qualitative Relations

Simple route descriptions typically use a rather restricted set of *action terms* such as "turn north", "turn right", etc. In the situations depicted in Figure 1, for example, one may describe the turn actions at the intersections by "straight on" and "turn sharp left", respectively.

Such egocentric relations can be defined in the Euclidean plane if one considers, for example, oriented points. An oriented point is a pair (p, o), where p represents the location of the point and o its absolute orientation. In geometrical terms, o is a half-line in the Euclidean plane that starts at point p (in our context the half-line corresponds to the line of sight of an agent). Relative to the oriented point, the plane can then be divided into different non-overlapping sectors each of which is labeled with an egocentric directional relation representing a turn action. More precisely, let $\mathcal{L}$ be a set of turn labels and let sec be a mapping that assigns to each oriented point (p, o) and each turn label $d \in \mathcal{L}$ a sector $\mathrm{sec}_{(p,o)}(d)$ of the plane. Each such sector is spanned by two half-lines starting in p. We require only that the set of all sectors $\mathrm{sec}_{(p,o)}(d)$ (with $d \in \mathcal{L}$) forms a partition of $\mathbb{R}^2 \setminus \{p\}$.

Interestingly, cognitive studies indicate that human conceptualizations of space are not based on equally sized sectors, not on symmetries with respect to front and back, and surprisingly not even on symmetries with respect to left and right (see, e.g., [Klippel and Montello, 2007]). This means that human conceptual schemes are often more irregular than the relational partition schemes that are usually considered in formalisms discussed in the Qualitative Spatial Reasoning field (see, e.g., [Renz and Nebel, 2007]). Nevertheless, our evaluation in Section 5 is based on egocentric variants of revised $\mathcal{STAR}^r$ calculi [Renz and Mitra, 2004], namely $\mathcal{STAR}_2^r$ (depicted left in Fig. 1), and $\mathcal{STAR}_4^r$ (right in Fig. 1), which takes into account empirical findings presented by Klippel and Montello [2007].

2.2 A Representation of Route Networks

In order to derive a qualitative representation of a route network, we assume that the metric information describing the layout of the network is given as a directed graph. Each vertex in the graph is associated with a position in the plane such that paths can accurately describe the shape of routes. We refer to such a graph as a *metric route network*.

Definition 1. A *metric route network* is an ordered triple $\mathcal{N} = \langle V, A, \varphi \rangle$, where $\langle V, A \rangle$ is a directed graph without loops and $\varphi \colon V \to \mathbb{R}^2$ is a function that assigns to each vertex v a point $v^{\mathcal{N}} := \varphi(v)$ in the plane.

In what follows we assume that $\mathcal{N}$ has no isolated vertices, i.e., each vertex occurs in at least one arc. Note that a metric route network is not necessarily planar or strongly connected. Metric route networks can be extracted from corpora of geodata that represent some kind of map, e.g., automatically recorded GPS traces [Edelkamp and Schrödl, 2003] or OpenStreetMap data. Further semantic annotations may provide additional information, for example, on the road type, the type of the point (e.g., intersection, landmark), and possible or permitted turn actions at an intersection.

While a metric route network represents a rich body of quantitative information, in order to generate or answer queries about route descriptions this level of detail is not needed. Firstly, depending on the application context not all parts of the metric route network are relevant for route descriptions, e.g., for car navigation only those parts of the metric route network are relevant that are accessible by cars due to width or traffic regulations. Secondly, for route descriptions the important vertices in the metric route network are those at which a decision has to be made by an agent traversing the network, i.e., these vertices have at least two outgoing arcs with regard to the application context.

Given a metric route network $\mathcal{N}$, we define: A *decision node* in $\mathcal{N}$ is a vertex with out-degree ≥ 2.[3] The set of decision nodes of $\mathcal{N}$ is denoted by $D_{\mathcal{N}}$. We assume that a subset of application-relevant decision nodes and paths is selected from $\mathcal{N}$. For $D \subseteq D_{\mathcal{N}}$, a *D-path* is a path $v_1 \dots v_n$ in $\mathcal{N}$ with $v_1, v_n \in D$ and $v_i \notin D$ for each $1 < i < n$ [Diestel, 2010]. $\Pi(D)$ denotes the set of all D-paths in $\mathcal{N}$.

We now define a qualitative representation of a metric route network that takes into account the egocentric perspective of an agent traversing the route network. The key idea is to explicitly represent the different ways in which the agent might

[3] In applications it can be useful to count also dead-ends as decision nodes.

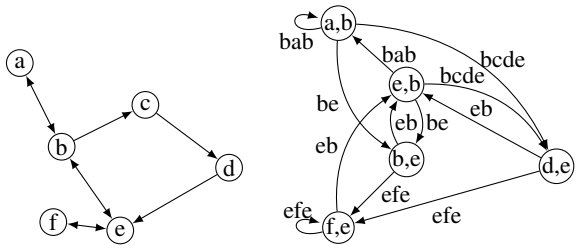

Figure 2: A metric route network and a decision frame depicted as multigraph (arcs are annotated with the paths they represent).

approach a decision node while traversing the route network. We refer to these ways as *states*. In each state the agent has options how it continues to traverse the network, i.e., there is a (potentially empty) set of possible decisions.

Given a fixed set of decision nodes D, each decision is associated with a unique D-path in the metric route network.

Since states model orientation, each decision can be properly labeled qualitatively: Given an arc (u, v_0) and a path $\pi = v_0\, v_1 \dots v_n$ in the metric route network $\mathcal{N}$, the *turn direction* for π with respect to (u, v_0) is the sector in which the path π continues relative to the incoming arc (u, v_0). More precisely, we set: $\text{turn}_{(u,v_0)}(\pi) = d$ if and only if $p_1 \in \text{sec}_{(p_0,o(q,p_0))}(d)$, where $q = u^{\mathcal{N}}$, $p_i = v_i^{\mathcal{N}}$ $(i = 0, 1)$, and $o(q, p_0)$ denotes that half-line starting in p_0 that can be extended to a straight line through q.

Definition 2. For a metric route network $\mathcal{N} = \langle V_{\mathcal{N}}, A_{\mathcal{N}}, \varphi_{\mathcal{N}} \rangle$, a *decision frame* on $\mathcal{N}$ is an ordered tuple $\mathcal{S} = \langle S, A \rangle$, where

- S is a nonempty set of arcs $(u, v) \in A_{\mathcal{N}}$ with $v \in D_{\mathcal{N}}$ (elements of S are called *states*). $D_{\mathcal{S}} := \{v : (u, v) \in S\}$ denotes the set of decision nodes of $\mathcal{S}$.
- $A\colon S \to 2^{\Pi(D_{\mathcal{S}})}$ is a function that assigns to each $(u, v) \in S$ a set of $D_{\mathcal{S}}$-paths π starting in v.

The elements of A are referred to as *decisions* and those of $A(s)$ as *decisions in state* s. We write $\|A\| := \max_{s \in S} |A(s)|$ for the maximal number of decisions in any state of $\mathcal{S}$. For $\pi = v \dots u'\, v' \in A(s)$, the arc (u', v') is in S (since π is a $D_{\mathcal{S}}$-path) and called the *successor state* of s under π.

Given a set of turn labels $\mathcal{L}$, a *labeled decision frame* $\langle S, A, l \rangle$ is a decision frame $\langle S, A \rangle$ augmented by a partial function $l\colon S \times A \to \mathcal{L}$ that assigns to each state $s \in S$ and each $D_{\mathcal{S}}$-path $\pi \in A(s)$ the turn direction for π relative to s, i.e., $l(s, \pi) = \text{turn}_s(\pi)$. A *path* in $\mathcal{S}$ is a sequence of states $\sigma = s_0 \dots s_n$ in S such that for each $1 \le i \le n$, s_i is a successor state of s_{i-1}. A *walk* in $\mathcal{D}$ is an alternating sequence $s_0\, d_1\, s_1 \dots d_n\, s_n$ of states and turn directions such that for each $1 \le i \le n$, s_i is a successor state of s_{i-1} under some decision π_i with $l(s_{i-1}, \pi_i) = d_i$.

Each walk can be described by the sequence of its turn labels. Correspondingly, we define route descriptions.

Definition 3. A *route description* $\Delta = d_1 \dots d_n$ is a sequence of labels from $\mathcal{L}$.

Decision frames as introduced here can also be cast as directed multigraphs in which each arc is annotated with a decision (cf. Figure 2). They are abstractions of the line graph [Diestel, 2010] of the metric route network (cf. the notion of evaluation mapping used in [Duckham and Kulik, 2003]). Moreover, it can be shown that labeled decision frames are an egocentric variant of the qualitative route graphs presented in [Renz and Wölfl, 2010].

In the following we require an *agent model* that defines how route descriptions are interpreted.

3 Agent Models

In many approaches the quality of a route description is determined as some sum of the values for the individual instruction primitives occurring in the description. The value of primitives in turn may depend on the particular location within the route network, but is otherwise independent of the overall graph. Obviously this has benefits for the computational cost of generating and evaluating descriptions. For such measures an optimal route can be found by a shortest path algorithm with an appropriate cost function. Haque *et al.* [2006] further evaluate route descriptions by measuring the success of an agent following it. We propose to generalize this idea by taking into account different types of agents that interpret instructions differently. In what follows we define a model that makes explicit how an agent interprets descriptions.

It is clear that there is wide range of agent models that could be defined, e.g., agents that use landmark information or that learn the route network they traverse. Here, we focus on some agent models with the following basic assumptions, namely: (i) the agent can follow a description regardless of its length, i.e., it does not forget or confuse parts of the instruction; (ii) the agent's interpretation of the turn direction coincides with the intended meaning of the direction; in case of ambiguities the agent is indifferent between the options associated with the same turn label; (iii) the environment is unknown to the agent and hence wrong turns cannot be recognized; (iv) the agent processes a route description step-by-step from the beginning, i.e., the agent tries to follow the first unprocessed label in the description (as long as there is one left) at the current state in the decision frame.

In order to model different types of simple agents, we consider the following features. We say that an agent (a) *strictly* processes a route description if it stops at the first state in which the next unprocessed instruction is not executable; (b) processes a route description with *default straight actions* if in case the current action cannot be executed, it tries to continue on some arc in straight direction, deferring the execution of the current action (if no path continues straight it has to stop); (c) *recognizes* the destination if it stops once a desired destination has been visited; (d) *learns* the visited states if it stops once a state is revisited, i.e., the agent performs loop checking.

Condition (b) expresses that a turn instruction is interpreted as "turn at the next opportunity", (c) expresses that an agent has some knowledge about the goal destination that allows him to stop processing a route description in goal states, and (d) expresses that an agent accumulates some knowledge about the traversed network such that it is certain to be lost if it arrives at a known place with the same previous orienta-

tion. These different features can be formalized using pushdown automata, but for the sake of simplicity we stick to the informal characterization of agents presented here.

Given an agent and some start state, a *(partial) execution trace* for a route description is a path in the decision frame traversed by the agent following a prefix of the description. An execution trace is said to be *complete* if the agent processes each instruction in the route description until it stops. The *stop set* of a route description, then, is the set of last visited states in all complete execution traces.

Lemma 1. Let $\mathcal{S} = \langle S, A, l\rangle$ be a labeled decision frame, $\sigma = s_0 \dots s_n$ a path in $\mathcal{S}$, $\Delta = d_1 \dots d_m$ a route description, and $S_\star$ a set of goal states in $\mathcal{S}$.

(a) σ is an execution trace for an agent processing Δ strictly iff $n \leq m$ and $s_0\, d_1\, s_1 \dots d_n\, s_n$ is a walk in $\mathcal{S}$.

(b) σ is an execution trace of Δ for an agent with default straight actions iff there exists a walk $s_0\, d'_1\, s_1 \dots d'_n s_n$ in $\mathcal{S}$, $m' \leq m$, and $1 \leq i_1 < \dots < i_{m'} \leq n$ such that $d'_{i_j} = d_j$ for all $1 \leq j \leq m'$ and for each $1 \leq i \leq n$ with $i \neq i_1, \dots, i_{m'}$, $d'_i =$ straight and there is no decision $\pi \in A(s_i)$ with $l(s_i, \pi) = d_{j_{\text{next}}}$, where $j_{\text{next}} = \min\{j : i_j > i\}$.

(c) σ is an execution trace for an agent with goal recognition iff $s_i \notin S_\star, 0 \leq i < n$.

(d) σ is an execution trace for an agent with learning iff $s_0 \dots s_{n-1}$ is a simple path.

Let $\text{tr}(s_0, \Delta)$ denote the set of execution traces starting in s_0 given Δ. It will be clear from the context which agent model the set is based on. In the remainder of the paper, we consider only agents based on Lemma 1 that are either processing strictly or perform default straight actions, where points (c) and (d) are optional further conditions. The next lemma follows from Lemma 1.

Lemma 2. Given a path σ in $\mathcal{S}$, a route description Δ, and start state s_0, the decision problem whether $\sigma \in \text{tr}(s_0, \Delta)$ can be decided in time $\mathcal{O}(|\sigma| \cdot \|A\|)$ for each considered agent model.

Naturally, all execution traces and the stop set can be derived by a recursive evaluation. The next section gives computational complexity bounds for several decision problems.

4 Queries on Decision Frames

We are interested in queries on decision frames that relate route descriptions and paths. To this end, we investigate the computational complexity in dependency of the agent model. For polynomial-time reductions, we construct $\mathcal{S}$ directly, since for any such frame we can find a suitable metric route graph $\mathcal{N}$ as a basis for $\mathcal{S}$.

4.1 Relating Route Descriptions to Paths

Theorem 1. The decision problem whether for given states $s_0, s_n \in S$ and route description Δ, there exists an execution trace $\sigma \in \text{tr}(s_0, \Delta)$ ending in s_n can be decided in time $\mathcal{O}(|S| \cdot |\Delta| \cdot \|A\|)$ for any considered agent without learning. For any of our learning agents the problem is NP-complete.

Proof Sketch. For agents without learning, one can apply a dynamic programming approach based on a table with $|S| \cdot |\Delta|$ entries. In case of a strict agent, one needs to check $\mathcal{O}(\|A\|)$ decisions in $\mathcal{S}$ for each entry. In case of an agent with default straight actions, possible deferred actions need to be evaluated as well. However, this can be handled by a table of double size, where each entry also exists with a prefixed straight action – the bound of $\mathcal{O}(\|A\|)$ still applies.

For agents with learning, the proof idea can be sketched as follows: NP membership follows immediately from the fact that n is bounded by $|S|$ and Lemma 2. NP-hardness can be obtained by a polynomial-time reduction of the Hamiltonian path problem, which is NP-complete even in cases of planar graphs [Garey *et al.*, 1976]. We construct for a given undirected graph G a suitable $\mathcal{S}$. For this we insert a state for each vertex and add an extra start and goal state such that the start state has a decision leading to any state except the goal state. Each undirected arc is represented with two (directed) decisions. These decisions are labeled with a non-straight action d_1. We further add decisions from each state to the goal state, but label them with a different action d_2. A fixed route description $\Delta = d_1 \dots d_1\, d_2$ with as many d_1 instructions as nodes in G forces an agent to visit all nodes (otherwise it has to stop early). Hence Δ leads from start to goal state iff there is a Hamiltonian path in G. □

Theorem 1 gives the complexity of testing whether a given description and initial state can lead to a given destination. The natural extension is to derive the entire stop set.

Theorem 2. Let $stop(s_0, \Delta)$ be the stop set of route description Δ starting in state $s_0 \in S$. Given initial state $s_0 \in S$ and route description Δ, the decision problem whether $\{s_1, \dots, s_n\} = stop(s_0, \Delta)$, is (1) polynomial-time for our non-learning agents, and (2) D^P-complete (= BH_2-complete [Cai *et al.*, 1988]) for our learning agents.

Proof Sketch. (1) follows from Theorem 1, since the stop set can be derived in the same manner. For (2), note that testing $\{s_1, \dots, s_n\} \subseteq stop(s_0, \Delta)$ is trivially NP-complete, and thus its complement is co-NP-complete. Therefore the decision problem is in $D^P = \mathsf{BH}_2$. For hardness, consider the SAT–UNSAT problem (a tuple of formulas (φ_1, φ_2): is φ_1 SAT and φ_2 UNSAT?) which is D^P-complete [Papadimitriou and Yannakakis, 1984]. We can equivalently think of such a tuple as two undirected graphs and the Hamiltonian path problem. From Theorem 1 it follows that for both we can construct a decision frame and a route description such that a description leads to the goal state iff there is a Hamiltonian path. Given two such labeled decision frames $\mathcal{S}_1, \mathcal{S}_2$ and Δ_1, Δ_2 we can combine the decision frames using suitable padding (i.e., a simple path) before $\mathcal{S}_1$ and starting in the $\mathcal{S}_2$ goal state such that we obtain $\mathcal{S}'_1$ and $\mathcal{S}'_2$, where the execution traces of the concatenation of Δ_2 and Δ_1 are the same as in Δ_1 in $\mathcal{S}_1$ and Δ_2 in $\mathcal{S}_2$. Using two new states and a new action label d, we can combine the decision frames such that the description prefix $d\, d$ allows the agent to terminate in any state except the goal states of $\mathcal{S}'_1, \mathcal{S}'_2$ (denoted $S_\star$) or leads it to the start states of $\mathcal{S}'_1, \mathcal{S}'_2$. Thus, (φ_1, φ_2) is in SAT–UNSAT iff $S \setminus \{S_\star\} = stop(s_0, d\, d\, \Delta_2\, \Delta_1)$ on the decision frame. □

4.2 Relating Paths to Descriptions

We are further interested in finding a description of a given path in $\mathcal{S}$. We show the stronger result for bounded length.

Theorem 3. The decision problem whether for a given path $\sigma = s_0 \dots s_n$ and bound $k \in \mathbb{N}$, there exists a description Δ of σ with $|\Delta| \leq k$ is polynomial-time for all our agents.

Proof. For learning agents we can first establish whether σ is simple. We cast σ as a graph and add arcs corresponding to the effect of decisions and, if applicable, default straight actions (restrictions by σ and Lemma 1 apply). A shortest path search for s_n yields the desired result. □

4.3 Shortest Descriptions

As our last formal result, we show that optimizing description length for a given start and destination state is a non-trivial task for agents featuring learning combined with default straight actions.

Theorem 4. The decision problem whether for given states $s_0, s_n \in S$ and bound $k \in \mathbb{N}$, there exists a route description Δ with $|\Delta| \leq k$ such that s_n is in the stop set $stop(s_0, \Delta)$ is (1) polynomial-time for all our agents without learning in combination with default straight actions, and (2) NP-complete for all our other agents.

Proof Sketch. In case (1), we can cast $\mathcal{S}$ as a graph, augment it with straight actions if applicable, and perform shortest path search to find the shortest description. It induces a simple path if no default straight actions are used. With such default actions, a shortest description might induce only non-simple paths in the decision frame, which is not a problem since the agent is not learning. In case (2), we can easily observe that the problem is in NP, since we can trivially guess (and describe) a suitable path adhering to the bound (cf. Theorem 3). For hardness, MinimumSetCover [Karp, 1972] can be encoded as a suitable decision frame. Due to space limitations, we can only roughly sketch the construction. Given $B = \{b_1, \dots, b_l\}, B_i \subseteq B$, and the decision problem $\exists I : \bigcup_{i \in I} B_i = B, |I| \leq k$, the bound on description length can be used to capture the original bound on the number of sets, and the learning property can be exploited to guarantee the cover. Essentially, we have as subframes trees for every B_i with two leaves (one include, one reject). Default straight actions can be exploited such that the cost of reject is low and that of include rather high. Reject leaves lead to paths with "$b_i \in B_j$"-states for all i with $b_i \in B_j$. Note, these will be *blocked* if the reject leaf is traversed. We connect the trees sequentially and add at the end an evaluation matrix over i, j such that it can only be traversed along i if there is at least one path through "$b_i \in B_j$"-states. Note, that this corresponds to the cover condition, as there is a path iff for each b_i there is some unblocked "$b_i \in B_j$"-state. Finally, we add between each reject leaf and the next subframe a path of dummy states such that the path can be traversed cheaply with a default action, but leads from all states to a dead end with action d. The same d is used in the evaluation matrix such that any path through $\mathcal{S}$ that exits a B_j-tree directly from a "$b_i \in B_j$"-state into the evaluation matrix has high cost. □

	$\mathcal{STAR}_2^r$	$\mathcal{STAR}_4^r$
a^s	17.83 (7.14)	17.83 (7.14)
a^{dg}	10.06 (4.04)	9.96 (4.19)
a^{dgl}	10.05 (4.04)	9.96 (4.19)

Table 1: Average and standard deviation of the length of route descriptions optimized for different agent models in different qualitative representations.

$\mathcal{STAR}_2^r$

Opt.\Eval.	a^s	a^{dg}	a^{dgl}
a^s	0.41 (0.38)	0.41 (0.38)	0.41 (0.38)
a^{dg}	0.01 (0.11)	0.25 (0.33)	0.24 (0.33)
a^{dgl}	0.02 (0.12)	0.29 (0.35)	0.29 (0.35)

$\mathcal{STAR}_4^r$

Opt.\Eval.	a^s	a^{dg}	a^{dgl}
a^s	0.62 (0.36)	0.62 (0.36)	0.62 (0.36)
a^{dg}	0.01 (0.11)	0.56 (0.39)	0.56 (0.39)
a^{dgl}	0.01 (0.11)	0.61 (0.38)	0.61 (0.38)

Table 2: Probabilities (in $[0, 1]$) for reaching the destination with respect to different agent models used for optimizing description length (rows optimized for, column evaluated on).

5 Evaluation

We report on a first empirical evaluation of the agent models based on a metric route network of Canberra taken from OpenStreetMap covering approximately 20 km^2 with 14 837 vertices and 4 445 arcs. The decision frame takes into account all streets and decision nodes in the map. It has 1 719 states and 4 934 decisions. The aim of the study is to measure the influence of the chosen agent model and the qualitative representation on evaluation and generation of route descriptions.

In the following we use the $\mathcal{STAR}_2^r$ and $\mathcal{STAR}_4^r$ representation schemes as outlined in Section 2.1. As agent models we consider: (a) only strict (a^s), (b) default straight actions and goal recognition (a^{dg}), and (c) default straight actions, goal recognition and learning (a^{dgl}). Based on a fixed randomly generated set of 1 000 distinct start-destination pairs, we generated shortest route descriptions for these agent models and evaluated each such description on all agent models. Despite the computational complexity of the evaluation for some of these models, it was feasible in all cases – most likely due to the low average degree of real-world route networks. In the case of a^{dgl} we had to resort to a common branch-and-bound search for shortest descriptions. The search took less than half a second for any pair.

Our results show that with regard to description length, there are only small differences between the considered qualitative representations and also between routes optimized for a^{dg} and a^{dgl} (cf. Table 1). Naturally the shortest path without default actions is unaffected by the qualitative representation.

As a second test, we consider the probability of reaching the destination when an agent (with agent model a) follows a description that has been optimized for an agent model a' Note that the agent model imposes a uniform distribution on decision choices, which induces a probability distribution over paths. Table 2 shows that assuming default straight actions for the generation of route descriptions usually reduces the probability of reaching the destination. This is plausible

because descriptions can rely on many (potentially ambiguous) *implicit* straight actions. However, it can also be seen that the more refined qualitative representation with a smaller "straight" sector mitigates this effect. We observed that the average number of distinct paths to the goal for a^{dgl} is about 1.14-1.38 for $\mathcal{STAR}_2^r$ and 1.04-1.064 for $\mathcal{STAR}_4^r$. For a^{dg}, infinitely many paths to the destination might exists in cases where loops in the network may be traversed arbitrarily often.

Further, we found that optimized descriptions for a^{dg} can also be reasonably used by a^{dgl}-agents. For only 6.9% of the descriptions in $\mathcal{STAR}_2^r$ (1.9% for $\mathcal{STAR}_4^r$) no simple path was induced.

6 Conclusion and Future Work

In this paper we have introduced a simple qualitative representation useful for evaluating and optimizing qualitative route descriptions. Further, we have identified several natural features of agents interpreting such descriptions, which play a crucial role for the analysis which paths agents traverse given a description. Identifying such paths enables a number of interesting criteria that can be used to evaluate and optimize route descriptions: Examples include the chance of reaching the destination, the expected distance to the destination, or the set of possible places where agents could end up.

For elementary queries about relations between paths and descriptions and also for optimizing description length, we have provided computational complexity bounds in dependency of the agent model. Several of our results indicate that it is (theoretically) infeasible to find optimal descriptions for certain criteria. Especially the identification of possible end states of routes under a fixed description is already hard for some simple agent models. If future evaluations show that for specific agent models the generation of optimal descriptions is in fact not practically feasible (e.g., optimizing the chance of reaching the destination), it will be necessary to analyze possible approximations. The results of our work indicate that it is important to balance a number of criteria when optimizing routes in order to guarantee robust descriptions for different underlying route networks and different agents processing them.

Acknowledgments

The authors thank the anonymous reviewers for helpful feedback on earlier drafts of this paper and Robert Mattmüller for fruitful discussions on complexity issues. This work was supported by DFG (*SFB/TR 8 Spatial Cognition*, project R4-[LogoSpace]), an ARC Future Fellowship (FT0990811), and a joint Go8/DAAD project (D/08/13855).

References

[Cai *et al.*, 1988] Jin-Yi Cai, Thomas Gundermann, Juris Hartmanis, Lane A. Hemachandra, Vivian Sewelson, Klaus Wagner, and Gerd Wechsung. The Boolean hierarchy I: Structural properties. *SIAM Journal on Computing*, 17(6), 1988.

[Diestel, 2010] Reinhard Diestel. *Graph Theory (Fourth Edition)*. Springer, 2010.

[Duckham and Kulik, 2003] Matt Duckham and Lars Kulik. "Simplest" paths: Automated route selection for navigation. In *COSIT*, LNCS 2825, pages 169–185. Springer, 2003.

[Duckham *et al.*, 2010] Matt Duckham, Stephan Winter, and Michelle Robinson. Including landmarks in routing instructions. *J. Location Based Services*, 4(1):28–52, 2010.

[Edelkamp and Schrödl, 2003] Stefan Edelkamp and Stefan Schrödl. Route planning and map inference with global positioning traces. In *Computer Science in Perspective*, LNCS 2598, pages 128–151. Springer, 2003.

[Garey *et al.*, 1976] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976.

[Haque *et al.*, 2006] Shazia Haque, Lars Kulik, and Alexander Klippel. Algorithms for reliable navigation and wayfinding. In *Spatial Cognition*, LNCS 4387, pages 308–326. Springer, 2006.

[Karp, 1972] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[Klippel and Montello, 2007] Alexander Klippel and Daniel R. Montello. Linguistic and nonlinguistic turn direction concepts. In *COSIT*, LNCS 4736, pages 354–372. Springer, 2007.

[Krieg-Brückner *et al.*, 2004] Bernd Krieg-Brückner, Udo Frese, Klaus Lüttich, Christian Mandel, Till Mossakowski, and Robert J. Ross. Specification of an ontology for route graphs. In *Spatial Cognition*, LNCS 3343, pages 390–412. Springer, 2004.

[Kuipers, 2000] Benjamin Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119(1-2):191–233, 2000.

[Mark, 1986] David M. Mark. Automated route selection for navigation. *Aerospace and Electronic Systems Magazine*, 1(9):2–5, 1986.

[Papadimitriou and Yannakakis, 1984] Christos H. Papadimitriou and Mihalis Yannakakis. The complexity of facets (and some facets of complexity). *Journal of Computer and System Sciences*, 28(2):244–259, 1984.

[Renz and Mitra, 2004] Jochen Renz and Debasis Mitra. Qualitative direction calculi with arbitrary granularity. In *PRICAI*, LNCS 3157, pages 65–74. Springer, 2004.

[Renz and Nebel, 2007] Jochen Renz and Bernhard Nebel. Qualitative spatial reasoning using constraint calculi. In *Handbook of Spatial Logics*, pages 161–215. Springer, 2007.

[Renz and Wölfl, 2010] Jochen Renz and Stefan Wölfl. A qualitative representation of route networks. In *ECAI*, Frontiers in Artificial Intelligence and Applications 215, pages 1091–1092. IOS Press, 2010.

[Richter and Duckham, 2008] Kai-Florian Richter and Matt Duckham. Simplest instructions: Finding easy-to-describe routes for navigation. In *GIScience*, LNCS 5266, pages 274–289. Springer, 2008.