# Chapter 6

# Global Mapping: Minimal Route Graphs Under Spatial Constraints

The matching of AGVGs as described in the previous chapter works well as long as the annotated metric information is reliable. This makes the approach well suited for identifying correspondences in AGVGs perceived in short order. However, while the robot moves around, the uncertainty, for instance in the position estimates of the nodes, accumulates and can grow without bounds. As a result of this and due to the fact that our matching algorithm currently does not bridge between different parts of the map AGVG, correctly closing cycles in the graph which correspond to large loops in the environment becomes difficult.

Therefore, a global mapping framework has to be built on top of the AGVG matching, which is the topic of this chapter. Our approach is to deal with the global mapping and loop closing problem by focusing on determining the correct discrete graph topology, relying on coarse but dependable spatial information instead of relying on the uncertainty-afflicted concrete metric annotations. The idea is to first determine the correct high-level graph structure using a multi-hypothesis tracking approach to deal with the uncertainty at the topological level. A concrete (H)AGVG can then be derived from a specific hypothesis.

As a consequence of this idea, we here regard the global mapping problem as the problem of determining the correct topology of a graph-like environment from a sequence of observations and interpret it as the task of finding a minimal route graph model that is consistent with the observations. The minimal model finding formulation of the mapping problem directly leads to a multi-hypothesis approach in which multiple consistent route graph hypotheses are tracked simultaneously.

The problem of exploration and map learning in graph-like environments has been investigated by several authors (Bender et al., 1998; Dudek et al., 1991, 1996, 1997; Rekleitis et al., 1999). Kuipers et al. (Kuipers, 1985; Kuipers & Byun, 1991) describe a *rehearsal procedure* to verify if two observed nodes can correspond only based on node signature information. This procedure involves moving to known neighbored

J.O. Wallgrün, *Hierarchical Voronoi Graphs: Spatial Representation and Reasoning for Mobile Robots*, DOI 10.1007/978-3-642-10345-2_6, © Springer-Verlag Berlin Heidelberg 2010

nodes and actively matching the neighborhoods up to a given distance. Dudek et al. (1991) point out that without further information successful map learning cannot be guaranteed. They also show that for undirected graphs a single movable marker is sufficient and that exploration requires $O(mn)$ edge traversals for a graph with $m$ nodes and $n$ edges. In Dudek et al. (1996), a passive map-learning approach is described in which a tree of all possible graph models is maintained.

Kuipers proposed formulating topological mapping as an abductive learning problem of finding the minimal model that explains a sequence of observations and actions (Kuipers et al., 2004; Remolina & Kuipers, 2004). In Kuipers et al. (2004), a tree of all models consistent with the axioms of the SSH is maintained in a way similar to the approach of Dudek et al. (1996). The simplest model is given by a prioritized circumscription policy. Places are derived from local metric maps and compared using descriptions of the local topology. The number of models in the tree can grow exponentially with the number of performed actions.

Planarity of the mapped environment has been exploited in the context of marker-based exploration in Rekleitis et al. (1999) and in the context of abductive map learning in Savelli & Kuipers (2004). In both cases, the result is a significant increase in the computational efficiency of the respective approaches.

In this work, we investigate the minimal model learning approach from the perspective of spatial consistency and as an application of spatial reasoning techniques developed in the area of qualitative spatial reasoning. An overview on the relevant concepts and techniques from this area of research is provided in Appendix B. Our focus is on direction information and we are especially interested in how different kinds of direction information (in the form of different qualitative spatial calculi) affects the size of the space of hypotheses which are consistent with the given information. In addition, we consider planarity as a constraint and require that the route graph models be embedded in the plane without crossing edges. We study the problem in a branch and bound search framework based on the estimated solution size, which results in a further reduction of the explored search space.

We proceed by first investigating the problem of finding the minimal route graph model in the simplified theoretical setting of a general graph world. Later, we adapt the developed approach to the Voronoi-based representations described in this book.

## 6.1 Theoretical Problem

Consider the following problem: A robot is roaming through a graph-like environment like the one shown in Fig. 6.1. The environment consists of junctions and straight hallways connecting the junctions. Whenever the robot arrives at a junction, it observes and memorizes a set of leaving hallways with some additional (qualitative) direction information. The direction information can either be absolute with regard to a given reference direction (e.g., “corridor x branches off to the north”) or relative (e.g., “corridors x and y meet at an obtuse angle”). We will call a description of such a perception

of a junction a *junction observation*, and define it as follows.

**Definition 6.1** (Junction observation)**.** *A junction observation is given by a triple* $J = (L, succ, R)$ *where*

- $L$ *is a set* $\{l_1, l_2, ..., l_m\}$ *of pairwise different elements from a set* $\mathcal{H}$ *of hallway identifiers, one for each perceived leaving hallway,*
- $succ : L \rightarrow L$ *is a total function specifying the immediate successor in the counterclockwise cyclic order in which the leaving hallways are perceived, and*
- $R$ *is a spatial description induced by the directions of the leaving hallways in* $L$ *with regard to a given set of direction relations.*

Given a junction observation $J$, we will write $L(J)$, $s(J)$, and $R(J)$ to refer to the respective elements of the triple $J$. We will not further define the nature of the spatial description $R$ here because it depends on the particular formalism used to describe the directions of the leaving hallways. An example could be a description using cardinal direction relations $n, nw, w$, etc. which consists of unary relation tuples, e.g., $n(l_1)$, $sw(l_2)$, and so on. The description could also involve binary relations holding between pairs of leaving hallways. Using cardinal directions, the second junction observation $J_2$ from the example in Fig. 6.1 could be given by

$$J_2 = (\ \{l_1, l_2, l_3, l_4\},\ \text{succ}(l_i) = l_{(i \oplus 1)},\ \{n(l_1), w(l_2), s(l_3), e(l_4)\}\ )$$

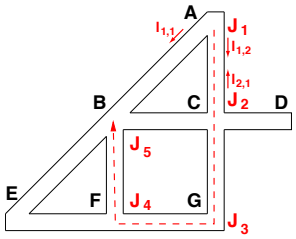
Later in this chapter, we will need to specify the distance of two objects $x$ and $y$ (either leaving hallways in a junction observation or edges incident to a particular node) with regard to the cyclic order defined by a successor function succ. For this purpose, we define the following distance function:

$$d_{\text{succ}}(x, y) = k \iff y = \text{succ}^k(x) \ \wedge\ \nexists l < k : y = \text{succ}^l(x) \tag{6.1}$$

$\text{succ}^k(x)$ here stands for the $k$-times composition of successor function succ with itself and, hence, the $k$th successor of $x$ in the cyclic order. For instance, for $J_2$ we get $d_{\text{succ}}(l_2, l_4) = 2$ and $d_{\text{succ}}(l_3, l_2) = 3$.

Junction observations are connected by *hallway traversal* actions consisting of leaving the current junction via one of the observed leaving hallways and arriving at the next junction via one of the leaving hallways belonging to the next junction observation.

**Definition 6.2** (Hallway traversal)**.** *A hallway traversal is described by a quadruple* $T = (J_s, l_s, J_e, l_e)$ *where* $J_s$ *and* $J_e$ *are junction observations with* $J_e \neq J_s$ *and* $l_s$ *and* $l_e$ *are leaving hallway identifiers with* $l_s \in L(J_s)$ *and* $l_e \in L(J_e)$*. It describes the action of taking leaving hallway* $l_s$ *after observing* $J_s$ *and arriving via* $l_e$ *of the next junction observation* $J_e$*.*




$\langle\ J_1 = (\{l_{1,1}, l_{1,2}\}, \text{succ}_1(x), R_1),$ // 1st junction observation
$T_1 = (J_1, l_{1,2}, J_2, l_{2,1}),$ // 1st hallway traversal
$J_2 = (\{l_{2,1}, l_{2,2}, l_{2,3}, l_{2,4}\}, \text{succ}_2(x), R_2),$ // 2nd junction observation
$T_2 = (J_2, l_{2,3}, J_3, l_{3,1}),$ // 2nd hallway traversal
$J_3 = (\{l_{3,1}, l_{3,2}\}, \text{succ}_3(x), R_3),$ // ...
$T_3 = (J_3, l_{3,2}, J_4, l_{4,1}),$
$J_4 = (\{l_{4,1}, l_{4,2}, l_{4,3}\}, \text{succ}_4(x), R_4),$
$T_5 = (J_4, l_{4,2}, J_5, l_{5,1}),$
$J_5 = (\{l_{5,1}, l_{5,2}, l_{5,3}, l_{5,4}\}, \text{succ}_5(x), R_5)\ \rangle$

Figure 6.1: The robot walks through a simple environment passing five junctions (dashed arrow). On the right is the history of junction observations and hallway traversal actions made by the robot

We will call $J_s$ the *start observation* and $J_e$ the *end observation* of the hallway traversal $T$ and use functions startj($T$), starth($T$), and endj($T$), endh($T$) for referring to the respective elements of hallway traversal $T$.

A list of alternating junction observations and hallway traversals corresponding to a walk of the robot through the graph will be referred to as a *history*.

**Definition 6.3** (History)**.** *A history $\mathcal{H}$ is a list $\langle J_1, T_1, J_2, T_2, ..., T_{n-1}, J_n\rangle$ of pairwise different junction observations $J_i$ and pairwise different hallway traversals $T_j$ with*

- $\forall i, j, 1 \leq i \leq n, 1 \leq j \leq n, i \neq j : L(J_i) \cap L(J_j) = \emptyset$
- $\forall i, 1 \leq i \leq n-1 : \mathit{startj}(T_i) = J_i \wedge \mathit{endj}(T_i) = J_{i+1}$

While junction observations directly correspond to physical junctions in the environment, a leaving hallway identifier $l \in L(J)$ does not correspond to a physical hallway in our formalization but rather to a "directed hallway" leading away from the junction corresponding to the observation $J$. Hence, we demand that the leaving hallway identifiers used in the individual junction observations be disjoint in the definition above. Figure 6.1 shows on the right the complete history corresponding to the walk given by the dashed arrow. Each $\text{succ}_i$ function here is given by $\text{succ}_i(l_{i,j}) = l_{i,(j \oplus 1)}$. In the following, we will use the notation $\mathcal{JO}_\mathcal{H}$ for the set of junction observations contained in histories $\mathcal{H}$ and $\mathcal{HT}_\mathcal{H}$ for the contained hallway traversals. In addition, $\mathcal{LH}_\mathcal{H}$ stands for the union of all $L(J)$ for all $J \in \mathcal{JO}_\mathcal{H}$ and, thus, contains all leaving hallway identifiers used.

Histories can be displayed as acyclic graphs, which we will call *history graphs* in the following and which will serve as the starting point for our actual minimal model finding algorithm. A history graph can be seen as the *maximal* route graph model of the environment that explains the history (maximal without adding completely unrelated nodes). This means that no junction observations or leaving hallways are unified except when they definitely have to correspond to the same physical junction because the robot just backtracked along previously traversed hallways. History graphs consist of two kinds of nodes: nodes that stand for junctions which have been observed and

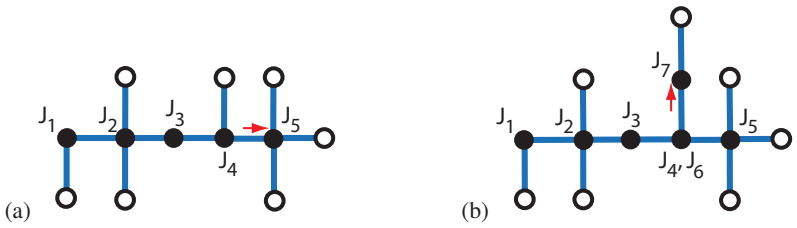



Figure 6.2: Representation of a history as an acyclic graph: Black nodes depict visited places, while white nodes stand for unvisited ones. (a) The graph corresponding to the history given in Fig. 6.1. (b) The resulting graph if the robot would continue by moving back from junction $B$ to junction $F$ and then turning right into the hallway leading to $E$

nodes that stand for the not yet observed end points of leaving hallways that have not been traversed.

Figure 6.2 shows on the left the history graph for the history from Fig. 6.1. Black nodes stand for the observed junctions and white nodes for the unobserved ones. As no backtracking took place, each black node stands for a single junction observation and each edge connecting two black nodes corresponds to a hallway traversal.

The right picture in Fig. 6.2 shows the graph that would result from continuing the walk by moving back to junction $F$ (resulting in junction observation $J_5$) and then moving to $E$ (junction observation $J_7$). As moving back to $F$ means that the robot moves back along the last traversed hallway, $J_4$ and $J_6$ *have* to correspond to the same junction and, hence, have to be unified in every valid route graph model derived from the history. As a result, the history graph remains unchanged by the traversal action. In contrast, continuing the walk by moving to $E$ results in assigning $J_7$ to the corresponding white node, changing it into a black node, and adding new edges and white nodes for the new leaving hallways (one in this case).

Following this construction scheme, complementing the history graph based on a hallway traversal and junction observation pair takes constant time and, hence, the complete construction takes $O(n)$ time for a history of length $n$ ($n$ hallway traversals and $n+1$ junction observations). The history graph representation is not strictly equivalent to the definition of a history though, as the exact number and order of traversal actions and observations cannot be retrieved without further information being stored. However, when annotated with the direction information included in the junction observations and with the final position of the robot and the last traversed edge, it contains all the information needed to transform it into smaller route graph models that are still valid explanations of the history. When depicting history graphs or smaller route graph models derived from the history graph, we will typically not provide the direction relations explicitly, but in most cases place the nodes in compliance with the direction

information.

Given a particular history, we are now interested in the problem of finding the minimal route graph model that explains the sequence of observations and traversal actions. As explanations we consider route graph models together with mappings from the history to walks through the hypothetical graphs. Route graph model here refers to an undirected graph together with a combinatorial embedding similar to the EGVGs defined in Chap. 3 but without loops or parallel edges, as we restrict ourselves to environments consisting of straight hallways. In order to avoid confusion with the nodes in the search tree discussed later in this chapter, we will refer to the nodes and edges of a route graph hypothesis explicitly as *route graph nodes* (RGNs) and *route graph edges* (RGEs), respectively.

A hypothesis (route graph model and mapping) is a valid explanation of the history if the following holds: We can draw the RGNs of the combinatorial embedded route graph into the plane with all RGEs being non-crossing straight line segments so that the hypothetical walk through this route graph model would then reproduce the history. This condition can be split into three classes of constraints that have to be satisfied in order to offer a valid explanation for a given history:

- **Structural constraints:** Under the term structural constraints we subsume all constraints that are not linked to the actual assignment of coordinates to the RGNs by the drawing: (1) The hypothetical walk through the given route graph hypothesis needs to reproduce the sequence of leaving hallway numbers of the original walk. (2) The cyclic order of leaving hallways needs to match the cyclic order of leaving edges of the corresponding nodes. (3) The distance between the arriving and leaving hallway in the cyclic order of leaving hallways needs to match for each passing of a node.

- **Planarity constraint:** The combinatorial embedding given by the cyclic edge orderings needs to be planar and the drawing must be a straight-line drawing of this embedding.

- **Direction constraints:** For each passed RGN, the drawing of the graph into the plane needs to induce the same direction relations for the leaving hallways as described in the junction observations.

To provide a more formal definition of these constraints, we first introduce the concept of a *map hypothesis*, which captures the structural constraints as these can be directly applied when constructing a hypothetical route graph model from a given history. We then define a map hypothesis to be *consistent* when the planarity constraint and direction constraints are satisfied. As already indicated previously, a map hypothesis not only consists of a route graph model but also contains a description of how the perceived junction hallways of the history map to the RGNs and RGEs of the route graph model. This mapping defines the hypothetical walk through the route graph that corresponds to the history.

**Definition 6.4** (Map hypothesis)**.** *Given a history $\mathcal{H} = \langle J_1, T_1, J_2, T_2, ..., T_{n-1}, J_n \rangle$, a map hypothesis $M_\mathcal{H}$ for a $\mathcal{H}$ is a triple $(G, mj, ml)$, where*

- *$G = (V, E, O)$ is an undirected graph $(V, E)$ with node set $V = \{v_1, v_2, ..., v_n\}$, edge set $E \subseteq V \times V$, and combinatorial embedding given by a set $O = \{succ_{v_1}, succ_{v_2}, ..., succ_{v_n}\}$ of successor functions of which $succ_{v_i}$ specifies the cyclic order of edges incident to $v_i$,*
- *$mj : \mathcal{JO}_\mathcal{H} \to V$ is a total function mapping the junction observations from $\mathcal{H}$ to nodes of $G$, and*
- *$ml : \mathcal{LH}_\mathcal{H} \to E$ is a total function mapping the leaving hallway identifiers from $\mathcal{H}$ to edges of $G$,*

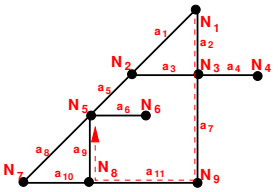
*that satisfies the following conditions:*

- $\forall J \in \mathcal{JO}_\mathcal{H} : (\forall l \in L(J) : mj(J) \in ml(l))$ *(incidence preservation)*
- $\forall T \in \mathcal{HT}_\mathcal{H} : ml(startl(T)) = ml(endl(T)) = \{startj(T), endj(T)\} \in E$
  *(traversal-edge mapping)*
- $\forall J \in \mathcal{JO}_\mathcal{H} : |L(J)| = \deg(mj(J))$ *(degree match)*
- $\forall J \in \mathcal{JO}_\mathcal{H}, s = s(J) : \big(\forall l \in L(J) : succ_{mj(J)}(ml(l)) = ml(s(l))\big)$
  *(cyclic order preservation)*
- $\forall i, 1 < i < n, s = s(J_i) :$
  $d_s(endl(T_{i-1}), startl(T_i)) = d_{succ_{mj(J_i)}}(ml(endl(T_{i-1})), ml(startl(T_i)))$
  *(cyclic order distance preservation)*

The first two conditions of the definition above simply ensure that junction observations and their leaving hallways are mapped to incident RGNs and RGEs and that the leaving hallways of a hallway traversal are mapped to the same RGE. The other three conditions are the structural constraints mentioned earlier: matching numbers of leaving hallways, preservation of the cyclic order information, and preserved distance between the arriving hallway and the leaving one in the cyclic order. We introduce the function $G(M)$ for accessing the route graph model $G$ in a map hypothesis $M = (G, \text{mj}, \text{ml})$.

Figure 6.3 shows four map hypotheses for the history given in Fig. 6.1. Each hypothesis is depicted by one drawing of the route graph into the plane. All examples preserve the combinatorial embedding of the hypothesis but not all consist only of straight lines and avoid crossing edges. The walk resulting from the mapping of the history to the route graph is shown by the dashed arrows. Disregarding the direction information, the walks resulting from the mappings would in all cases correctly reproduce the history of observations and hallway traversals.

(a)




$\mathrm{mj}(J_1) = N_1, \mathrm{ml}(l_{1,1}) = a_1, \mathrm{ml}(l_{1,2}) = a_2,$

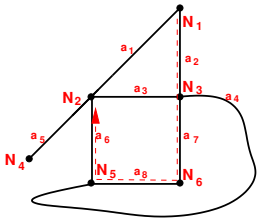
$\mathrm{mj}(J_2) = N_3, \mathrm{ml}(l_{2,1}) = a_2, \mathrm{ml}(l_{2,2}) = a_3, \mathrm{ml}(l_{2,3}) = a_7, \mathrm{ml}(l_{2,4}) = a_4,$

$\mathrm{mj}(J_3) = N_9, \mathrm{ml}(l_{3,1}) = a_7, \mathrm{ml}(l_{3,2}) = a_{11},$

$\mathrm{mj}(J_4) = N_8, \mathrm{ml}(l_{4,1}) = a_{11}, \mathrm{ml}(l_{4,2}) = a_9, \mathrm{ml}(l_{4,3}) = a_{10},$

$\mathrm{mj}(J_5) = N_5, \mathrm{ml}(l_{5,1}) = a_9, \mathrm{ml}(l_{5_2}) = a_6, \mathrm{ml}(l_{5,3}) = a_5, \mathrm{ml}(l_{5,4}) = a_8$

(b)




$\mathrm{mj}(J_1) = N_1, \mathrm{ml}(l_{1,1}) = a_1, \mathrm{ml}(l_{1,2}) = a_2,$

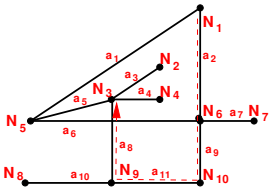
$\mathrm{mj}(J_2) = N_3, \mathrm{ml}(l_{2,1}) = a_2, \mathrm{ml}(l_{2,2}) = a_3, \mathrm{ml}(l_{2,3}) = a_7, \mathrm{ml}(l_{2,4}) = a_4,$

$\mathrm{mj}(J_3) = N_6, \mathrm{ml}(l_{3,1}) = a_7, \mathrm{ml}(l_{3,2}) = a_8,$

$\mathrm{mj}(J_4) = N_5, \mathrm{ml}(l_{4,1}) = a_8, \mathrm{ml}(l_{4,2}) = a_7, \mathrm{ml}(l_{4,3}) = a_4,$

$\mathrm{mj}(J_5) = N_2, \mathrm{ml}(l_{5,1}) = a_6, \mathrm{ml}(l_{5,3}) = a_3, \mathrm{ml}(l_{5,4}) = a_1, \mathrm{ml}(l_{5,5}) = a_5$

(c)




$\mathrm{mj}(J_1) = N_1, \mathrm{ml}(l_{1,1}) = a_1, \mathrm{ml}(l_{1,2}) = a_2,$

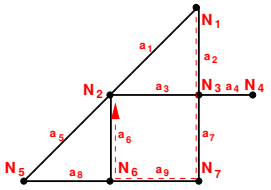
$\mathrm{mj}(J_2) = N_6, \mathrm{ml}(l_{2,1}) = a_2, \mathrm{ml}(l_{2,2}) = a_6, \mathrm{ml}(l_{2,3}) = a_9, \mathrm{ml}(l_{2,4}) = a_7,$

$\mathrm{mj}(J_3) = N_{10}, \mathrm{ml}(l_{3,1}) = a_9, \mathrm{ml}(l_{3,2}) = a_{11},$

$\mathrm{mj}(J_4) = N_9, \mathrm{ml}(l_{4,1}) = a_{11}, \mathrm{ml}(l_{4,2}) = a_8, \mathrm{ml}(l_{4,3}) = a_{10},$

$\mathrm{mj}(J_5) = N_3, \mathrm{ml}(l_{5,1}) = a_8, \mathrm{ml}(l_{5,2}) = a_4, \mathrm{ml}(l_{5,3}) = a_3, \mathrm{ml}(l_{5,4}) = a_5$

(d)




$\mathrm{mj}(J_1) = N_1, \mathrm{ml}(l_{1,1}) = a_1, \mathrm{ml}(l_{1,2}) = a_2,$

$\mathrm{mj}(J_2) = N_3, \mathrm{ml}(l_{2,1}) = a_2, \mathrm{ml}(l_{2,2}) = a_3, \mathrm{ml}(l_{2,3}) = a_7, \mathrm{ml}(l_{2,4}) = a_4,$

$\mathrm{mj}(J_3) = N_7, \mathrm{ml}(l_{3,1}) = a_7, \mathrm{ml}(l_{3,2}) = a_9,$

$\mathrm{mj}(J_4) = N_6, \mathrm{ml}(l_{4,1}) = a_9, \mathrm{ml}(l_{4,2}) = a_6, \mathrm{ml}(l_{4,3}) = a_8,$

$\mathrm{mj}(J_5) = N_2, \mathrm{ml}(l_{5,1}) = a_6, \mathrm{ml}(l_{5,2}) = a_3, \mathrm{ml}(l_{5,3}) = a_1, \mathrm{ml}(l_{5,4}) = a_5$

Figure 6.3: Four possible map hypotheses for the history of Fig. 6.1, given by the depictions of their route graph models and the mappings mj and ml. Only the examples in (a) and (d) are consistent. The model in (d) is also a minimal route graph model; it corresponds to the original environment

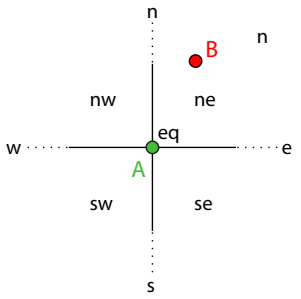



Figure 6.4: Relations from the cardinal direction calculus

Let us for now again assume that the spatial information is given in form of qualitative cardinal direction relations. The exact relations we are going to use are those of the cardinal direction calculus (Ligozat, 1998), a calculus for qualitative spatial reasoning which we are going to introduce more formally in Sect. 6.3.2.1 (see Appendix B for more details on qualitative spatial calculi). In the cardinal direction calculus, $n$, $w$, $s$, and $e$ correspond to specific angles, while $nw$, $sw$, $se$, and $ne$ comprise angle intervals as indicated in Fig. 6.4.

Storing the cardinal directions derived from compass readings for each leaving hallway of a junction observation results in the following spatial descriptions $R_i$ for our exemplary history from Fig. 6.1:

$$
\begin{aligned}
R_1 &= \{sw(l_{1,1}), s(l_{1,2})\} \\
R_2 &= \{n(l_{2,1}), w(l_{2,2}), s(l_{2,3}), e(l_{2,4})\} \\
R_3 &= \{n(l_{3,1}), w(l_{3,2})\} \\
R_4 &= \{e(l_{4,1}), n(l_{4,2}), w(l_{4,3})\} \\
R_5 &= \{s(l_{5,1}), e(l_{5,2}), ne(l_{5,3}), sw(l_{5,4})\}
\end{aligned}
$$

The drawing of the hypothesis of Fig. 6.3(a) shows that this hypothesis is consistent with the spatial information as it would correctly reproduce the cardinal directions for each junction observation. The drawing of the second hypothesis (Fig. 6.3(b)), however, is not consistent with the spatial information contained in the history: A spatial inconsistency arises because the hallway leaving from $N_5$ to the west has been connected with the one leaving $N_3$ to the east. As we assume straight hallways, it follows that $N_5$ has to be to the east of $N_3$. Locally, this information is consistent. However, as it is also known that $N_6$ is south of $N_3$ and $N_5$ is west of $N_6$, it follows that $N_5$ has to be in the southwest sector of $N_3$. This contradicts our previous conclusion that $N_5$ lies to the east of $N_3$. This reasoning shows not only that the depicted drawing is inconsistent with respect to the direction information, but also that no such drawing can exist for the given hypothesis. Hence, the entire hypothesis is inconsistent and does not offer a valid explanation of the history.

A second cause of inconsistency within a hypothesis is shown in the third example (Fig. 6.3(c)). Here the drawing is spatially consistent with regard to the direction information and it preserves the cyclic edge orders. However, it has crossing edges ($N_5$-$N_6$ and $N_3$-$N_9$) and, more importantly, no drawing without crossing edges exists because for this graph the combinatorial embedding is not planar.

These two inconsistent examples illustrate that the structurally valid map hypotheses can still be inconsistent with the available information or the underlying assumption that the environment is planar. This leads to the following definition of a consistent map hypothesis.

**Definition 6.5** (Consistent map hypothesis)**.** *A map hypothesis $M_{\mathcal{H}} = (G, mj, ml)$ is consistent if there exists a straight-line drawing $D(G)$ of $G$ into the plane that satisfies the following conditions:*

- *for every node in $G$, the natural cyclic order of leaving edges in the drawing $D(G)$ corresponds to the cyclic edge order specified in the combinatorial embedding (cyclic order preservation),*
- *the line segments in $D(G)$ corresponding to the RGEs of $G$ do not cross each other (planar drawing), and*
- *for every junction observation $J \in \mathcal{JO}_{\mathcal{H}}$, the spatial description induced by $D(G)$ for node $mj(J)$ matches the spatial description one gets by replacing all $l \in L(J)$ with ml$(l)$ in description $R(J)$ (matching direction information).*

The general idea of the minimal route graph model approach to the global mapping problem is to prefer among all consistent map hypotheses the one that is minimal in terms of the number of RGNs in the route graph. We will call such a hypothesis a minimal route graph model.

**Definition 6.6** (Minimal route graph model)**.** *Given the set $\mathcal{CM}_{\mathcal{H}}$ of consistent map hypotheses for $\mathcal{H}$, $M \in \mathcal{CM}_{\mathcal{H}}$ is a minimal route graph model for $\mathcal{H}$ if and only if no $N \in \mathcal{CM}_{\mathcal{H}}$ exists with $|V(G(N))| < |V(G(M))|$.*

The last example in Fig. 6.3(d) shows a second consistent map hypothesis, the one that corresponds to the original environment. The number of RGNs is smaller than in the first example and no consistent hypothesis with even less RGNs exists for the given history. Hence, it is a minimal route graph model for this particular history.

Following the minimal route graph model approach, the problem of mapping an unknown environment becomes a combinatorial optimization problem of incrementally computing one or all minimal consistent map hypotheses from the observations gathered during exploration. One thing that makes this problem interesting and complex is the fact that it combines combinatorial aspects with questions of spatial consistency. While the space of structurally possible map hypotheses grows exponentially with the length of the history, the direction constraints and planarity restriction reduce the search space by allowing us to prune inconsistent branches.

The fact that the definition of a consistent map hypothesis is based on the existence of a drawing of the graph that satisfies certain conditions is a further indication of the complexity of the problem: Clearly, it it infeasible to consider all drawings as there are infinitely many straight-line drawings of a graph. General techniques like describing the constraints in a system of (geometric) equations are typically much too expensive computationally to be applicable.

On the positive side, there exist techniques that solve individual aspects of the problem. For instance, checking whether a graph with combinatorial embedding is planar—which means that a drawing without crossing edges exists that preserves the combinatorial embedding—can be done in $O(n)$ time. In addition, for every planar drawing there exists one with straight edges. Consequently, we can filter out many inconsistent hypotheses by discarding all hypotheses with non-planar embeddings.

With regard to additional spatial information, in our case the direction relations, techniques for checking the consistency of a set of qualitative spatial relations have been developed in the research field of qualitative spatial reasoning. These techniques allow for determining whether a given network of spatial constraints is satisfiable. Again, employing these techniques allows filtering out a substantial part of inconsistent hypotheses. The minimal route graph model problem provides an interesting testbed for these kinds of approaches as it benefits from expressive spatial formalisms for which the consistency problem can be solved efficiently. One of our goals in this work is to investigate the suitability of existing qualitative direction calculi for this kind of problem and identify potential need for further research in this area.

In the following, we describe an approach to determine a minimal route graph model based on individually enforcing the planarity constraint and the consistency of the direction information. As a result of the individual constraint checking, the approach is incomplete in the sense that it may not filter out all inconsistent map hypotheses. For instance, it could happen that there exists a drawing for a given map hypothesis that is planar and one that is compliant with the direction constraints but not one that is both. However, not finding all constraints is still preferable to not using the available information at all. We start by first looking at the combinatorial optimization problem and developing a best-first branch-and-bound-based search procedure to solve it. In the next section, we integrate planarity and spatial consistency checks into the framework. The results of the empirical evaluation of this approach for two spatial calculi are provided in Sect. 7.3.

## 6.2 Branch and Bound Search for Minimal Model Finding

In this section, we describe a solution to the minimal route graph model problem that consists of a branch and bound search through the search tree of possible associations of RGNs and, hence, junction observations. The search starts with the history graph and effectively folds the graph onto itself by unifying the RGNs with their corresponding junction observations. A lower bound estimate of the model size as implied by

the associations already made is used to efficiently guide the search towards a minimal model in a best-first manner. As a result, the approach effectively performs an A* search through the interpretation tree.

### 6.2.1 Search Through the Interpretation Tree

Deriving route graph hypotheses from a history of junction observations and hallway traversals is mainly a problem of correctly identifying junction observations that correspond to the same physical junction in the environment. Hence, similarly to approaches on data association we encountered in the previous chapter, a solution to the problem can be formulated as a search through the tree of possible associations, the interpretation tree. In the case of the data association problem we had two disjoint sets of objects, the data set and the model set. Here, however, we only have one set of objects, the RGNs. In principle, each RGN can be associated with multiple other RGNs, resulting in a partition of the set of junction observations into equivalence classes. We still end up with a tree-formed search space in which each level corresponds to matching one particular RGN and each edge corresponds to one particular matching if we make matching assignments in the following way (see also Fig. 6.6):

- At each level in the interpretation tree, the corresponding RGN can only be matched to objects already matched at a higher level. Hence, the fact that junction observations mapped to RGN $A$ and RGN $B$ should correspond to the same physical junction could be expressed by first assigning $A$ to 0 (new junction) and then at a lower level matching $B$ to $A$.
- When more than two RGNs should be unified (more than two junction observations correspond to the same physical junction), each RGN has to be matched with the one previously associated in the tree: Assuming $A$, $B$, and $C$ should be unified to form a single junction and are associated in this order in the tree, first $A$ is assigned to 0, then $B$ is assigned to $A$, and finally $C$ is assigned to $B$.

For the following discussion we extend the graph representation for histories and derived route graph hypotheses so that it allows us to describe the partial matchings corresponding to inner nodes of the interpretation tree. This is done by distinguishing RGNs in the graph depending on whether they have already been matched or not. Already matched RGNs are depicted by circles as before, black for visited ones and white for unvisited ones. Not yet assigned RGNs are depicted by black and white crosses instead (see Fig. 6.5).

An additional deviation from the standard interpretation tree is that an assignment can lead to multiple child nodes as there may exist multiple route graph hypotheses resulting from joining two RGNs. One could argue that this would not be required if one would match hallways instead of junctions, but as our hypotheses also comprises junctions that have not been observed and for which consequently the number of leaving hallways objects is unknown, we think the chosen approach is more adequate.

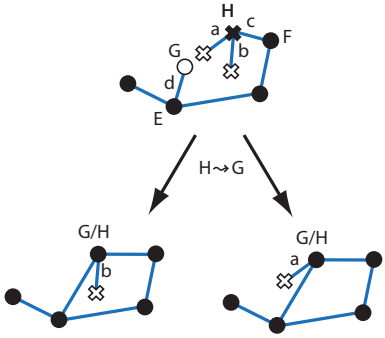



Figure 6.5: Assigning $H$ to $G$ results in two possibilities because the hallway $d$ can correspond to either $a$ or $b$. Therefore, this matching results in two successor nodes in the interpretation tree

Multiple successor nodes can arise when there exist multiple ways to map the leaving hallways of the matched junctions onto each other while preserving the cyclic order information. Figure 6.5 shows an example of such a situation: The picture shows at the top the partially constructed route graph hypothesis corresponding to a node in the search tree. We now assume that we want to identify RGN $H$ with RGN $G$. Since $G$ is an unvisited junction and $H$ is a visited one, the RGE $d$ of $G$ must correspond to one of the leaving RGEs of $H$, either to $a$ or to $b$. It cannot correspond to $c$ as RGNs $E$ and $F$ have already been assigned and, thus, cannot be unified anymore in this hypothesis. As a result two successor hypotheses are possible, shown below, and there would be two child nodes for this particular matching in the interpretation tree.

A complete search tree for a small example walk consisting of three junction observations is shown in Fig. 6.6. While in principle each RGN could be matched with any of the RGNs associated at a higher level, two junctions can only be connected by a single hallway, and matchings can directly imply other matchings. As a result, not all matchings occur in the interpretation tree depicted in the figure. Still, the number of map hypotheses grows exponentially with the length of the history.

As also illustrated in Fig. 6.6, we store the following information for each node in the search tree: a (partial) route graph hypothesis reflecting the partial matching and and an RGN list which contains all RGNs from the original history graph. For already matched RGNs the list states the assigned other RGN. For the still unassigned RGNs, the RGN list contains a list of matching candidates called its *match list*. Additional stored information not shown in the figure includes the mapping from the history to the junctions and hallways in the route graph model, given in the form of annotations to the RGNs and RGEs and the robot's current location within the route graph.

Several examples of node hypotheses are included in Fig. 6.6. $H_1$ is the original history graph, with all nodes depicted by crosses because no assignments have been made yet. As RGN $A$ is the first RGN considered, it has to be new, which results in

node hypothesis $H_2$. $B$, $C$, and $D$ are the end points of the leaving hallways of $A$ and, hence, have to be new junctions as well (their match lists only contain 0), which leads to hypothesis $H_5$. At the next level, $E$ can only be matched to $B$ or $C$ (or to 0) and not to $A$ because a connecting RGE between $A$ and $D$ already exists. $H_5$, $H_7$, $H_{14}$, and $H_{33}$ are intermediate node hypotheses along the marked path through the tree leading to $H_{83}$. As all nodes in $H_{83}$ are assigned, it is a complete map hypothesis. It is also the hypothesis that reflects the actual environment. The remaining examples are alternative complete map hypotheses of which $H_{47}$ assumes less junctions than $H_{83}$, while $H_{112}$ requires more. In $H_{83}$, it has been hypothesized that $C$, $E$, and $G$ correspond to a single physical junction and, as mentioned, this has to be realized by associating $E$ with $C$ before associating $F$ with $E$.

Overall, expanding a node in the interpretation tree during the search for the minimal model involves the following three steps: (1) The first still unassigned RGN in the RGN list is chosen; (2) the successor nodes for every matching of this RGN with a candidate in its match list are generated; (3) the successor nodes are added to a list containing the current fringe of the search tree. More details on the generation of successor nodes will be given in Sect. 6.2.3.2.

### 6.2.2 Best-First Branch and Bound Search Based on Solution Size

As it is our goal to find a minimal consistent map hypothesis in terms of the number of RGNs, upper and lower bounds on the model size over all hypotheses that can be generated from a particular node in the search tree can be used to decide whether an optimal solution can be contained in this part of the search space. Nodes with a lower bound higher than the currently found minimal upper bound can be completely excluded from the search. Hence, by employing a branch and bound search approach, we achieve a reduction of the search space.

In addition, a lower bound estimate can be used to efficiently guide the search towards a minimal model in a best-first manner by always expanding the node with the currently smallest lower bound. Once the chosen node with the minimal lower bound contains a hypothesis in which all nodes have been assigned, we have found a minimal map hypothesis.

Every inner node in the search tree contains a route graph model with still unassigned RGNs. Hence, it stands for a set of map hypotheses which can be generated by performing the remaining assignments. A lower bound on the number of RGNs contained in a map hypothesis in this set can be computed efficiently in the following way based on the node's RGN list: Every RGN that is assigned to 0 counts as one because it will occur as an RGN in any of the derived map hypotheses. The same holds for every still unassigned RGN for which the match list only contains 0 since this RGN cannot be matched with an already established RGN any longer. All other RGNs from the list have either been matched with an already counted RGN or their match lists still allow for such a matching. Hence, they do not count.

The resulting estimate is only a lower bound on the minimal number of RGNs

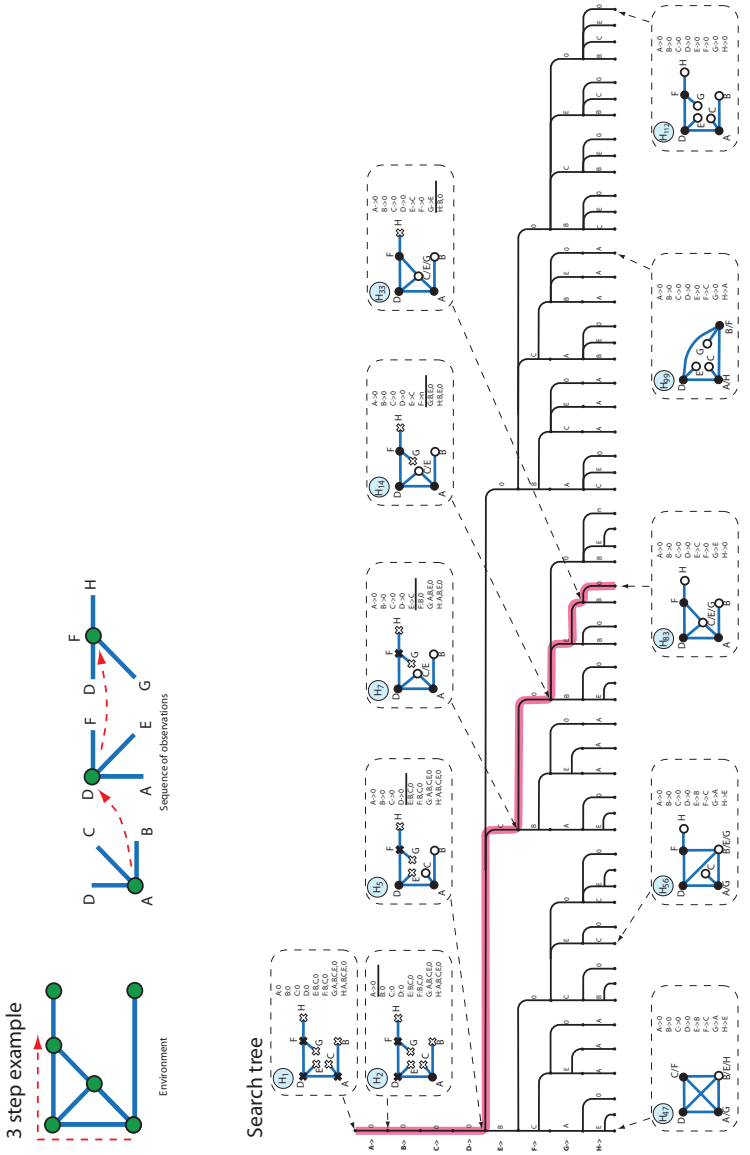



Figure 6.6: The complete interpretation tree resulting from a three-step walk through a simple environment. The environment and the sequence of observations are given at the top of the figure. The node data (graphs and RGN lists) is depicted for several nodes in the search tree

because later assignments may actually lead to cases in which the match list of an RGN is reduced to 0. This would result in an additional junction. However, to get a better lower bound would require us to perform a costly analysis of the dependencies in the match lists, while in our case we simply have to update some counters during the search.

In summary, the lower bound on the model size for a node in the search tree is defined as follows.

**Definition 6.7** (Model size lower bound)**.** *Given a node $n$ in the interpretation tree, the model size lower bound mslb$(n)$ of $n$ is defined as*

$$mslb(n) = N + O \tag{6.2}$$

*where $N$ is the number of RGNs assigned to 0 in the RGN list of $n$ and $O$ is the number of RGNs with 0 as the only element in their respective match lists.*

In the implementation of the search algorithm described here, the current fringe of the search tree is stored in a priority queue sorted by the nodes' lower bounds. In case two nodes have the same lower bounds, a secondary criterion and, if needed, a tertiary one are used to sort the nodes. The secondary criterion is the number of still unassigned RGNs in the RGN lists. As a consequence, nodes deeper in the tree and, hence, closer to a complete map hypothesis will be preferred. The tertiary criterion is the upper bound msub$(n)$ on the possible minimal model size for node $n$. It is computed by summing up the number of RGNs assigned to 0 and the RGNs that still need to be assigned and have 0 in their match list.

**Definition 6.8** (Model size upper bound)**.** *Given a node $n$ in the interpretation tree, the model size upper bound msub$(n)$ of $n$ is defined as*

$$msub(n) = N + P \tag{6.3}$$

*where $N$ is the number of RGNs assigned to 0 in the RGN list of $n$ and $P$ is the number of RGNs with 0 contained in their match list.*

To provide an example of the results of applying the best-first branch and bound search as described in this section, Fig. 6.7 shows at the top the original interpretation tree from Fig. 6.6 and in the middle the parts searched for a minimal model using the best-first branch and bound algorithm. The order in which the nodes are expanded is given by the numbers within the circles representing the nodes. The bottom figure shows the result of applying the same approach but searching until all minimal models have been found.

### 6.2.3 Expand and Update Operations

The main operation of the search procedure is the expansion of a node in which the child nodes based on all possible matchings are generated. However, in order to incorporate newly available history information without performing a new search from

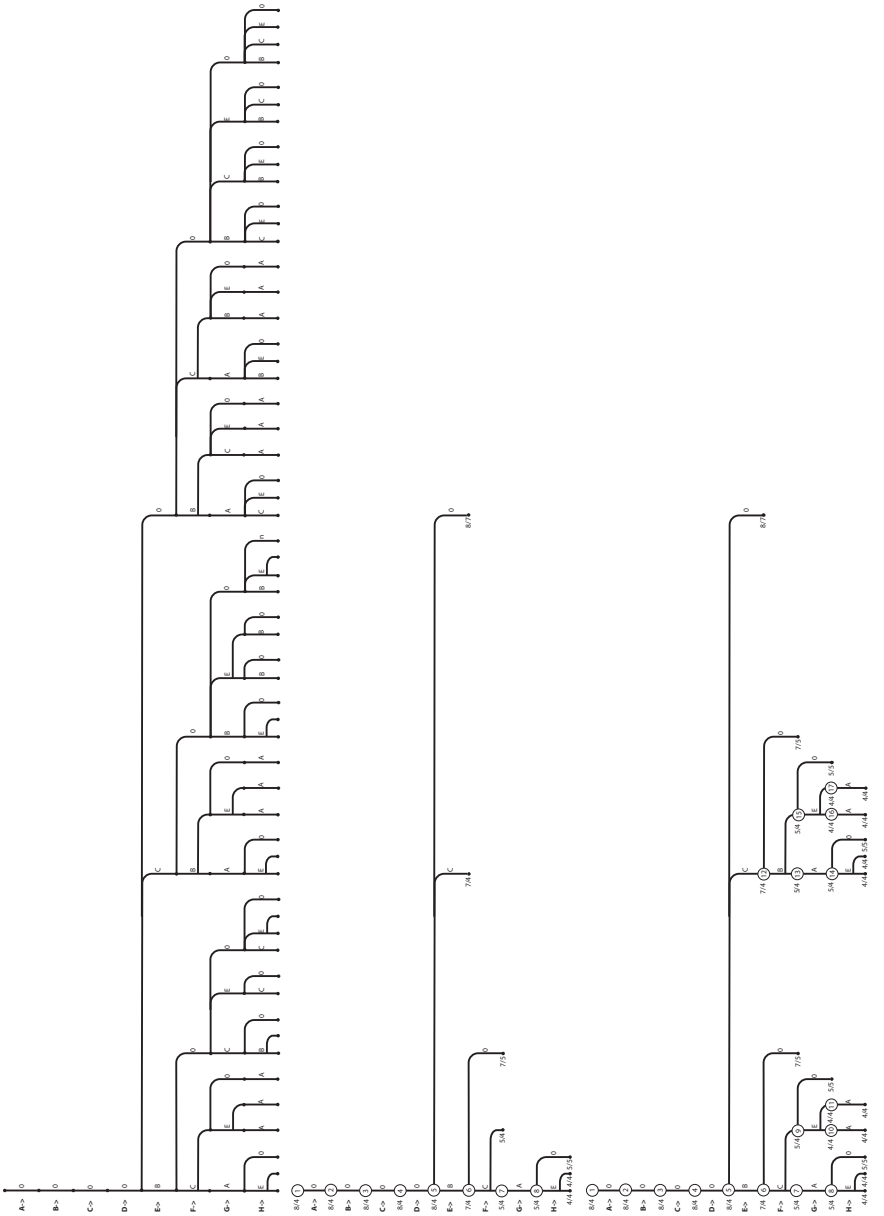


Figure 6.7: Effects of different versions of the minimal model algorithm (part 1): At the top the entire search tree, in the middle the tree searched with branch and bound for the first minimal model, and at the bottom the same searching for all minimal models

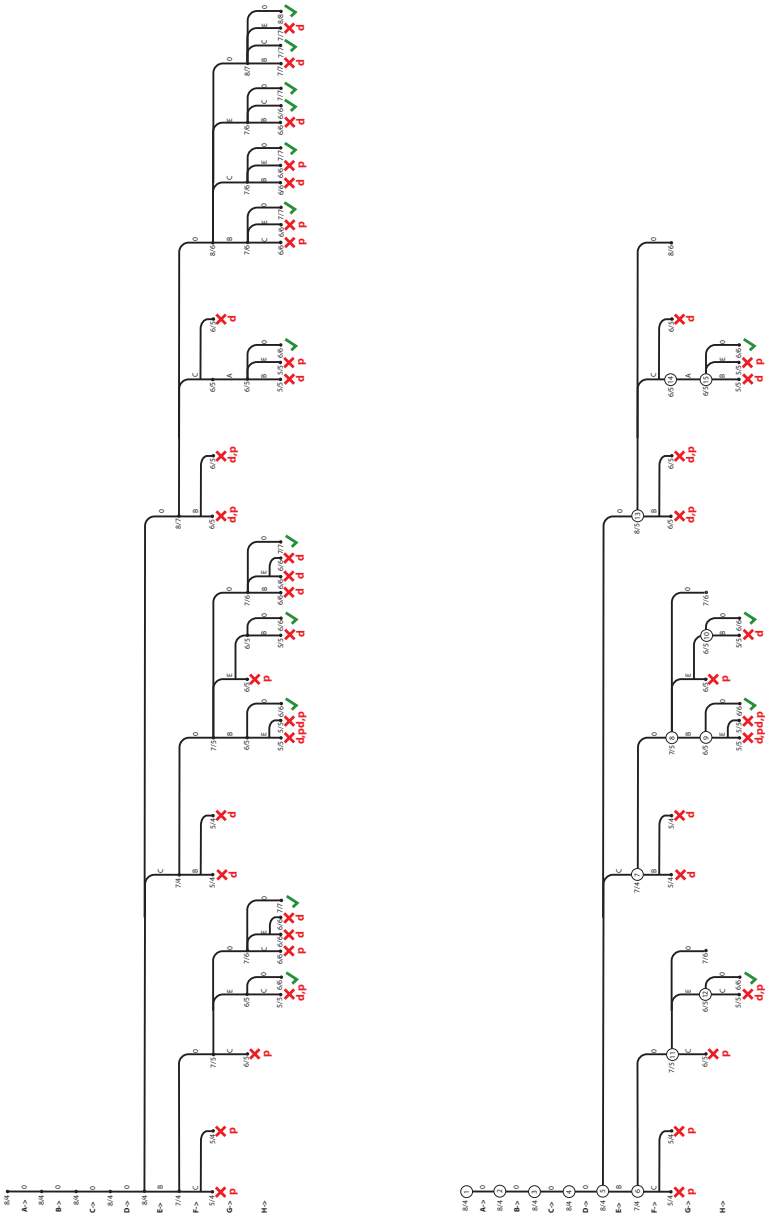


Figure 6.8: Effects of different versions of the minimal model algorithm (part 2): At the top the tree pruned by planarity and direction constraints and at the bottom the result of applying constraint-based pruning and branch and bound search

**Algorithm 7** Main loop of the minimal model finding algorithm

**procedure** minimalModel(History $\mathcal{H}$, PriorityQueue Q)

```
1:  minimalModelFound ← false
2:  while |Q| > 0 and not minimalModelFound do
3:      n ← pop(Q)
4:      if not uptodate(n) then
5:          L ← update(n, H)
6:          insert all elements from L into Q
7:      else if node has unassigned RGNs then
8:          L ← expand(n)
9:          insert all elements from L into Q
10:     else
11:         minimalModelFound ← true
12:         insert n into Q
13:     end if
14: end while
```

scratch, a second operation is required in which the node information is updated based on the new action and observation. We will refer to these two operations as the *expand* and *update* operations.

A pseudocode version of the main loop of the actual search procedure is shown in Algorithm 7. This procedure is called whenever new history information becomes available and terminates when a minimal model has been found. The current fringe of the search tree is provided in the form of the priority queue $Q$ sorted by criteria described in the previous section. In the main loop, the first element is taken from $Q$. It is then tested whether this node $n$ is up-to-date (meaning all history information has been incorporated) or not. If this is not the case, the update operation will be performed, which incorporates the next hallway traversal and junction observation. Like the expand operation, the update operation may result in multiple successor hypotheses. Therefore, it returns a list of new nodes which are inserted into $Q$.

If the node $n$ does not need to be updated, it is checked whether it still has unassigned RGNs. If this is the case, $n$ is expanded and successor nodes which are returned as a list are inserted into $Q$. Otherwise, $n$ has to be a minimal model and the search terminates. Before that, $n$ is put back into the queue so that we can continue with the search when new history information becomes available.

In the following, we consider the update and expand operations in more detail.

#### 6.2.3.1 Update Operation

The update operation updates a node $n$ based on one new hallway traversal $T$ and a new junction observation $J$. The result is a set of updated successor nodes. The following steps have to be performed:

1. A list $L$ is initialized as empty; in the end this list will contain the successor nodes.

2. The robot's location within the route graph hypothesis of $n$ is updated based on $T$.

3. Case 1: If the new location $loc$ is an already visited RGN, we check whether $J$ fits the current location. If this is the case, a copy $s$ of $n$ is added to $L$. Then mj$(J)$ is set to $loc$ and each ml$(l_i)$ for a leaving hallway in $J$ is set to the corresponding RGE in $s$. If $J$ and the visited RGN do not match, no successor nodes for $n$ will be created, effectively closing this branch of the interpretation tree.

4. Case 2: If the new location is an unvisited RGN, there may be multiple ways to map the already existing RGEs to the observed leaving hallways. Every existing RGE needs to be mapped to a different leaving hallway while the cyclic order needs to be preserved. For each valid mapping the following is done: A new node $s_i$ is constructed in which the graph has been updated accordingly. A new edge ending at a new unvisited RGN is attached at the right position in the cyclic order for each leaving hallway that does not correspond to an existing RGE. mj and ml are updated as in case 1 and the nodes $s_i$ are added to $L$.

5. For each node from $L$, new RGNs are added to the RGN list for each end point of the leaving hallways in $J$ (even though no actual RGN may have been added to the route graph model), and their match lists are set accordingly.

6. $L$ is returned as the result of the update operation.

To illustrate this procedure, two examples of update operations are depicted in Fig. 6.9. On the left, we see the current node hypothesis before the first update operation. All RGNs except $E$ are already matched. The new hallway traversal to be incorporated now leads the robot from $C$ to $B/D$ as indicated by the dashed arrow. A depiction of the new junction observation can be found between the arrows. It contains two more leaving hallways in addition to the arriving one. The end points of these will require the instantiation of two more RGNs, $F$ and $G$. $B/D$ is an unvisited RGN in the given hypothesis (case 2), which means that every existing RGE needs to correspond to an observed leaving hallway but not vice versa, and we already know that the RGE connecting $C$ and $B/D$ corresponds to the leaving hallway via which the robot arrived. Therefore, two mappings are possible, leading to the two new hypotheses at the end of the arrows. In the first one, $F$ is identified with $A$, and as a result its match list is set to $\{A\}$. For $G$ a new unvisited RGN is generated and connected to $B/D$ with a new edge. In the second hypothesis, $G$ and $A$ are identified, resulting in similar changes to the graph and RGN list.

Let us now assume, another hallway traversal is performed and both hypotheses are updated. Again, the traversed hallway is marked by the dashed arrows and the

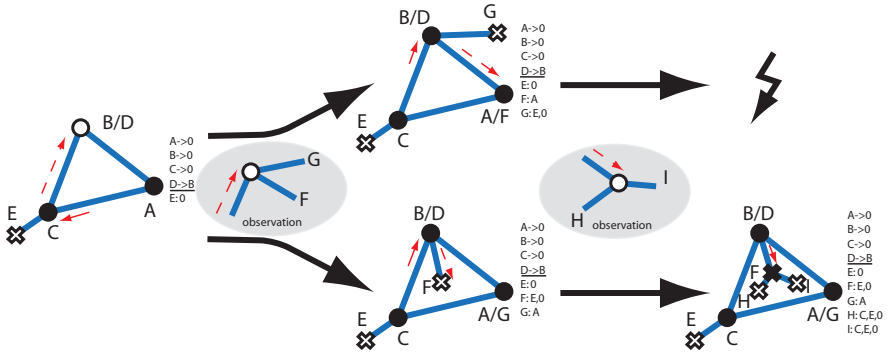



Figure 6.9: Example of two consecutive update operations resulting in one valid successor hypothesis

new observation is shown between the arrows. For the first hypothesis, the traversal would mean that the robot moves to $A/F$, which is a visited junction (case 1) with two RGEs overall. In this case, there must be a one-to-one mapping between existing RGEs and leaving hallways, which in this example is not possible as there are three observed leaving hallways. Therefore, the hypothesis is discarded. In contrast, updating the second hypothesis results in another instance of case 2, but here with only one possible mapping between RGEs and observed hallways. Hence, only a single updated hypothesis will be generated.

#### 6.2.3.2 Expand Operation

Expanding a node based on matching the next unmatched variable RGN $X$ with an RGN $W$ higher in the RGN list involves the generation of child nodes with modified graph structures and RGN lists for every valid way of folding the graph onto itself so that $X$ and $W$ are merged. The list of new child nodes is returned in the same way as it is by the update operation. In more detail, the expand operation performs the following steps:

1. A list $L$ is initialized as empty, which in the end will contain the successor nodes.
2. As $X$ and $W$ can both be visited or unvisited RGNs, four general cases of merging have to be distinguished. Typically, there are multiple possibilities of merging $X$ with $W$; and for some cases multiple ways of mapping the RGEs of $X$ to the RGEs of $W$ exist that preserve the cyclic order information. For every possible way of merging and edge mapping a new node $c_i$ is created and processed by the following steps:
    - The graph hypothesis is transformed according to the merging variant and edge mapping. For RGEs of $X$ that correspond to existing RGEs of $W$

the entire subtree attached to this edge needs to be matched recursively in accordance with the match lists of the involved RGNs. This can lead to additional possibilities, in which case the node $c_i$ is further split into multiple new ones, or to contradictions, in which case the hypothesis is discarded.

- $X$ is marked as matched to $W$ in the RGN list.
- For all unmatched RGNs that get merged in the recursive process, their match list is set to the RGN they are merged with because their matching is now determined as well.
- $W$ is removed from the match lists of the remaining unmatched variables.
- $c_i$ is added to $L$.

3. $L$ is returned as the result of the expand operation.

As mentioned, there are four general cases of merging, which we will not discuss in detail here. Instead, we will restrict ourselves to providing one rather complex example of matching a visited RGN to an unvisited one. We use the notation $A \rightleftharpoons B$ to refer to the RGE connecting the RGNs $A$ and $B$.

The starting hypothesis of our example can be seen at the top of Fig. 6.10. RGNs $A - F$ are already assigned; the others are unassigned. The variable to be matched in this expansion step is $G$ and we consider the matching with RGN $B$. As $B$ is unvisited, there are two possible merging variants, one in which RGE $A \rightleftharpoons B$ corresponds to RGE $G \rightleftharpoons I$ and one in which it corresponds to $G \rightleftharpoons H$.

In the first case, $G$ is merged with $B$ and the RGE $G \rightleftharpoons I$ is removed. The unvisited end node $I$ is merged with $A$ and its match list is changed accordingly. Finally, $B$ is removed from the remaining match lists and the new node is added to the successor list $L$.

In the second case, $H$ corresponds to $B$, but $H$ is a visited RGN with two more RGEs. As we are comparing two visited RGNs now, there has to be a one-to-one mapping between the RGEs so that RGE $H \rightleftharpoons K$ has to correspond to $A \rightleftharpoons D$ and $H \rightleftharpoons J$ to $A \rightleftharpoons C/F$. Both RGE pairs are merged in the following. As $I$ is only an unvisited end point, the recursion ends on this side. However, $J$ is a visited RGN which has to correspond to unvisited RGN $C/F$ and there are two possible edge mappings. This means that the current hypothesis has to be split into two new ones, one for each mapping. In the first one, $J \rightleftharpoons M$ is merged with $C/F \rightleftharpoons E$, in the second one, $J \rightleftharpoons L$ with $C/F \rightleftharpoons E$. In both cases, the recursion ends. Overall, we end up with three successor nodes for the original node depicted in the bottom row of the figure.

### 6.2.4 Two Variants of the Minimal Model Finding Problem

Up to now, we have described a version of the minimal model finding problem in which each model is a complete closed environment which might contain unvisited

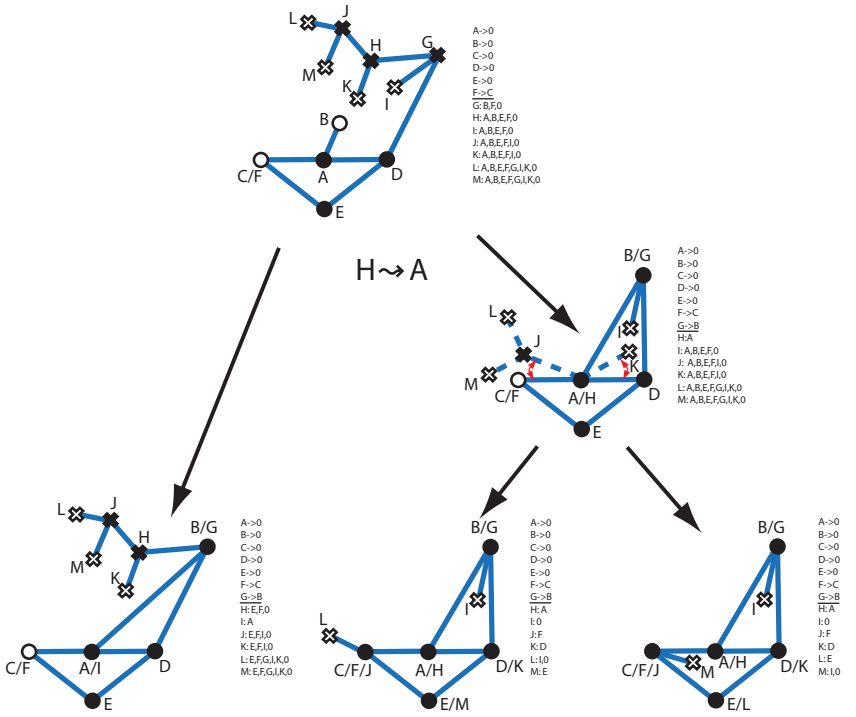



Figure 6.10: Example of matching $G$ to $B$ in an expand operation, resulting in three different child nodes

junctions that form the end points of perceived but never traversed hallways. A less complex version of the problem can be obtained by restricting the models to visited places and allowing hallways with open endings. This problem is less complex because the number of possible matchings is reduced significantly and because the lack of information about the unvisited junctions allows for more variations in general.

In the experimental analysis of this work, we will compare both variants of the algorithm. The version dealing with complete environments will be referred to as the *CompEnv* variant, while the version only determining the layout of the visited parts of the environment will be called the *VisOnly* variant.

The implementation of the VisOnly variant is simply a modified version of the CompEnv algorithm in which variables are only instantiated for junction observations and not for the end points of the leaving hallways. As a result, the search tree from Fig. 6.6 would be reduced to a simple linear chain of three edges as none of the visited RGNs can be joined. While this illustrates the reduced complexity of the VisOnly variant, it is an extreme case as the history only consists of three junction observations and no junction is visited twice.

Besides the complexity issues, the question of which variant is better suited in practice needs to be answered in the context of a concrete application scenario. The advantage of the CompEnv variant is that it includes a certain predictive power which can, for instance, be useful to predict shortcuts when applied to route networks like street networks or the hallway networks used as an example here. However, for more low-level graph abstractions like Voronoi graphs, CompEnv often leads to a higher number of wrong predictions until the entire environment has been explored.

After providing a solution to the purely combinatorial problem, we now turn to the question of how additional constraints, based either on planarity or on direction information, can be incorporated into the search algorithm.

## 6.3 Pruning Based on Spatial Constraints

As mentioned previously, our approach is to check planarity constraints and consistency of the direction information separately. We start with a discussion of the planarity constraint.

### 6.3.1 Checking Planarity

In this work, we are exclusively dealing with graph environments that are plane graphs. This means that they are embedded into the plane without crossing edges. This fact allows us to reduce the set of possible hypotheses. Each graph hypothesis for which the cyclic order information does not describe a planar embedding can be immediately discarded because such a graph cannot be drawn into the plane without crossing edges in a way that preserves the cyclic edge orders. The criterion for deciding whether a general graph with a combinatorial embedding describes a planar embedded graph is that its *genus* is 0. The genus of an undirected graph $G = (V, E)$ is given by Euler's formula:

$$\text{genus}(G) = (|E| + 2c - |V| - i - f)/2 \tag{6.4}$$

where $c$ is the number of connected components in the graph, $i$ is the number of nodes of degree 0 (isolated nodes), and $f$ is the number of faces formed by traversing edges in accordance with the cyclic ordering information (a more precise definition will be given below). We only consider connected graphs without isolated nodes here and thus the formula becomes

$$\text{genus}(G) = (|E| - |V| - f)/2 + 1 \tag{6.5}$$

Our approach to planarity checking is similar to the one described in Savelli & Kuipers (2004). First of all, it is advantageous to internally transform the undirected route graphs into bidirected graphs in which each RGE from the original graph is represented by a pair of edges with opposite directions. The information that $e$ and $f$ correspond to the same RGE and thus are *reversals* of each other ($\text{rev}(e) = f$ and

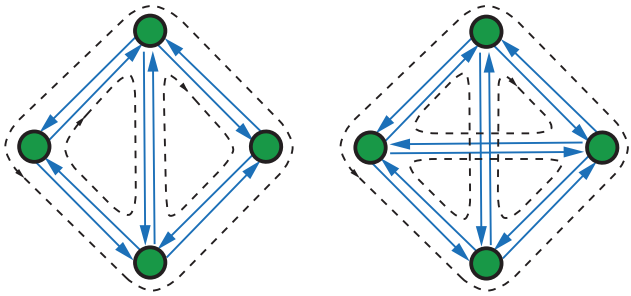


Figure 6.11: Two bidirected graphs of which the first has three faces and thus according to Eq. 6.5 depicts a planar embedding, while the second with an additional (undirected) edge has two faces, which means the embedding is not planar

rev$(f) = e$) is stored in the form of edge attributes[1]. In addition, the cyclic order information is transformed so that now each RGN $v$ is annotated with the cyclic order of directed edges which have $v$ as source. In the following, we assume that pred$(e)$ and succ$(e)$ yield the predecessor and successor edge of $e$ in the cyclic order of leaving directed edges at the source node of $e$. One can then define sequences of edges by a function next$(e)$ as follows:

$$\text{next}(e) = \text{succ}(\text{rev}(e)) \tag{6.6}$$

Based on this function, each directed edge now is part of exactly one cycle of directed edges $e_1, e_2, ..., e_n$ with $e_1 = e_n$ and $e_{i+1} = \text{next}(e_i)$. These cycles are called the *faces* of the graph. Figure 6.11 shows two combinatorial embedded graphs. Their faces are depicted by the dashed arrows. The left graph has three faces, while the right one with an additional (undirected) edge has only two. As a consequence, Eq. 6.5 yields that genus $= 0$ for the first graph and genus $= 1$ for the second graph. Hence, only the first depicts a planar combinatorial embedding.

Planarity checking can be performed in linear time (Hopcroft & Tarjan, 1974; Lempel et al., 1967). We integrate planarity checking into our search algorithm by representing the route graph hypotheses as bidirected graphs and updating the face information and pointers for pred, succ, and next whenever we modify the graph structure. As soon as Eq. 6.5 is violated, the hypothesis at hand can be discarded as the planarity constraint is violated. In our approach, the faces are numbered and each edge of the bidirected graph stores the number of the face it belongs to (given by facenumber$(e)$). The relevant operation which has the potential of changing planarity in our approach is inserting a new edge into the graph. After inserting an edge which results in the two new directed edges $e$ and rev$(e)$ and updating the successor information, three cases have to be distinguished, illustrated in Fig. 6.12:

[1]The resulting structure is often referred to as a "map" (Mehlhorn et al., 1999) but we will not use this term here in order to avoid ambiguities.

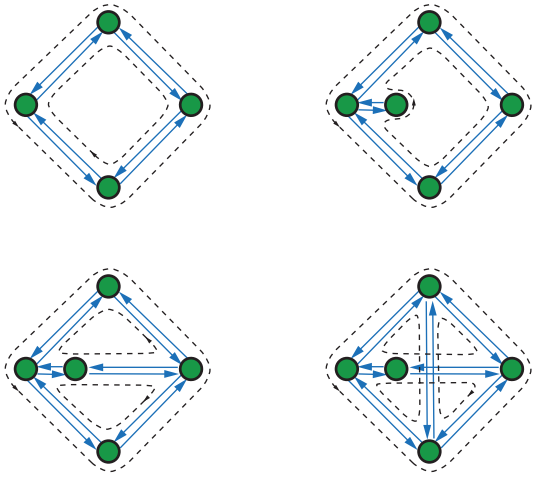


Figure 6.12: Three cases of modifying a graph, starting with the planar graph at the top left. Only in the last case (bottom right) does the genus change, and, hence, the resulting embedding is not planar

1. next$(e)$ = rev$(e)$: This is the case when we insert a new node and connect it to an old one. The number of faces stays the same while the number of nodes and edges increase by one, leaving the genus unchanged. The face numbers of both, $e$ and rev$(e)$ are set to the number of next(rev$(e)$).

2. facenumber(next$(e)$) = facenumber(next(rev$(e)$)): The new edge connects two already contained nodes and splits an existing face into two new ones. The total number of faces increases by 1. The face number of $e$ is set to the number of the old cycle, while rev$(e)$ gets a new number, and numbers of all edges belonging to the same face as rev$(e)$ are updated accordingly. Since the number of edges also increased by one and the node remained unchanged, the genus again remains unchanged.

3. facenumber(next$(e)$) $\neq$ facenumber(next(rev$(e)$)): In this case, the old face would be replaced by a new one that combines both faces, and hence the face number would decrease by 1. The edge number would increase by one, while the number of nodes remains the same. As a result, the genus would increase to 1. Since this means that the embedding is not planar anymore, the hypothesis can be immediately discarded.

While the first and third case require constant time, the second case takes linear time in the number of edges because of the need to update the face numbers for one face. We provide more details on the incorporation of the planarity check into the overall search algorithm in Sect. 6.3.3.

### 6.3.2 Checking Spatial Consistency

One of the main goals of the work described in this chapter is to investigate how the presence of spatial information provided in the form of qualitative direction relations that only represent coarse information but can be perceived reliably reduces the number of hypotheses that have to be considered. Checking the consistency of a route graph hypothesis with regard to spatial constraints stemming from the perceived directions of the leaving hallways requires us to determine whether an assignment of points in the plane to the RGNs of the hypothesis exists which induces the same set of relations.

Hence, we are faced with a constraint satisfaction problem in which the domain (points in $\mathbb{R}^2$) is infinite. However, research on qualitative spatial reasoning has produced constraint-based techniques to deal with this kind of problem. The solution typically consists of a qualitative constraint calculus defining a set of spatial relations and algebraic operations like converse and composition on the set of relations. Depending on the particular calculus, consistency checking can be performed by employing the so-called algebraic closure algorithm or a more involved backtracking search over the set of all possible scenarios which are then tested again by the algebraic closure algorithm. The algebraic closure algorithm requires $O(n^3)$ time, where $n$ is the number of related objects. We provide an overview on these concepts and techniques in Appendix B.

Based on these result, our approach is to formulate the perceived direction information in the form of qualitative direction relations from particular qualitative spatial calculi, derive a network of constraints from the given route graph hypothesis, and apply the standard consistency check methods to the constraint network. For performing the consistency check we use the spatial reasoning toolbox SparQ, which provides implementations of a large set of spatial calculi and the standard reasoning techniques (Wallgrün et al., 2006, 2007).

As we are particularly interested in comparing the effects of absolute and relative direction information on the search space and on the number of solutions, we have chosen the absolute cardinal direction calculus (Ligozat, 1998) and the relative $\mathcal{OPRA}_2$ calculus (Moratz, 2006; Moratz et al., 2005) for our analysis.

Both calculi cannot be considered as ideal, but no better candidates or other reasoning formalisms exist to our knowledge. Hence, the problem investigated here can also be seen as a challenge for qualitative spatial reasoning research. As an ideal calculus we would consider one with the following properties:

1. good computational properties with regard to the consistency check,
2. expressive enough to rule out many hypotheses,
3. dealing with relations that are easily and reliably accessible,
4. able to express the cyclic edge order information.

The cardinal direction calculus, on the one hand, is rather efficient as a large tractable subset exists for which the algebraic closure algorithm decides consistency (cf. Appendix B.4). This subset contains all relations required in our context. The downside is that the calculus does not allow for expressing the cyclic ordering information about the leaving RGEs in the route graph. As a result, it can happen that a constraint network deemed consistent by the consistency check only has solutions for which the cyclic order information is not preserved.

The $\mathcal{OPRA}_2$ calculus, on the other hand, can express the cyclic ordering information. However, employing the algebraic closure algorithm to $\mathcal{OPRA}_2$ constraint networks generated from our route graph hypotheses only results in an incomplete method to rule out inconsistent cases. This is also true if the much more inefficient backtracking search would be employed because algebraic closure does not decide consistency even if the constraints are all base relations. We still have chosen $\mathcal{OPRA}_2$ as to our knowledge no relative direction calculus exists with significantly better computational properties. In addition, the calculus offers a similar level of granularity as that of the cardinal direction calculus. This is beneficial for the comparison. How problematic the application of an incomplete consistency checking method is has to be evaluated experimentally.

In the next two sections, we describe how we model the direction information for both calculi.

#### 6.3.2.1 Modeling Spatial Configurations in the Cardinal Direction Calculus

The cardinal direction calculus is an absolute binary qualitative direction calculus describing the cardinal direction of one point object from another point object using the nine base relations we saw in Fig. 6.4. Here, we use the base relations from the calculus to describe the directions of leaving hallways as seen from the corresponding junction, but then transfer this information into a constraint over the possible positions of the connected junctions in the plane. As a result, each RGE in a route graph hypothesis yields a direction constraint between the connected RGNs.

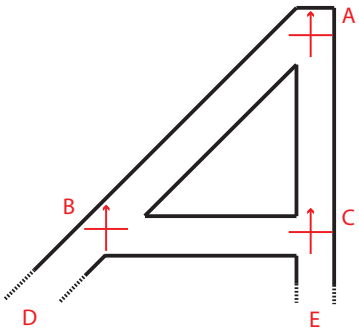
Absolute direction information like the cardinal direction information can actually be exploited in multiple ways in the minimal model finding algorithm. It can be used to enforce three different requirements:

1. **Valid direction orderings:** When adding a new RGE to an RGN, it can only be inserted into the cyclic edge order at a position where the edges also preserve the cyclic order of cardinal directions. For instance, a resulting cyclic order of edges with directions $n, s, w$ is not valid as $w$ would have to appear between $n$ and $s$.

2. **Valid junction matching:** When matching two RGNs, mappings of RGEs are only valid if corresponding RGEs have the same directions.

3. **Global consistency:** There needs to be a way of assigning coordinates to the RGN such that the direction constraints are satisfied.

As we explained, the global consistency check requires the full constraint reasoning approach based on the algebraic closure algorithm. Therefore, we first generate a constraint network from the given route graph hypothesis. This constraint network consists of one variable for each RGN and one constraint represented by a directed edge for each RGE. In Fig. 6.13, we see part of a route graph hypothesis and the corresponding set of derived constraints that make up the constraint network. For all other pairs of junctions, the constraint holding between them is the disjunction of all base relations except $eq$. The constraint network is then fed into SparQ, which performs the consistency check. If the algebraic closure algorithm discovers an inconsistency, the hypothesis at hand can be discarded.




| | | |
|---|---|---|
| A | ne | B |
| A | n | C |
| B | w | C |
| B | ne | D |
| C | n | E |

Figure 6.13: A route graph hypothesis and the cardinal direction constraints derived from it

#### 6.3.2.2 Modeling Spatial Configurations in the $\mathcal{OPRA}_2$ Calculus

The second calculus employed and investigated in this book is the $\mathcal{OPRA}_2$ calculus. In contrast to the cardinal direction calculus, it is a relative calculus describing the relative orientation of two objects to each other. Hence, a robot would only need to be able to estimate the angles between the leaving hallways and would not require a compass to determine $\mathcal{OPRA}_2$ relations.

$\mathcal{OPRA}_2$ is actually one particular instance of a calculus from the Oriented Point Relation Algebra ($\mathcal{OPRA}_m$) family. $m$ here is the granularity parameter used to determine the number of base relations that are distinguished (Moratz, 2006; Moratz et al., 2005). The domain of ($\mathcal{OPRA}_m$) is the set of oriented points (points in the plane with an additional direction parameter).

For a given granularity parameter $m \in \mathbb{N}$ the concrete set of $\mathcal{OPRA}_m$ relations is derived as follows: For each of the two related oriented points, $m$ lines are used to

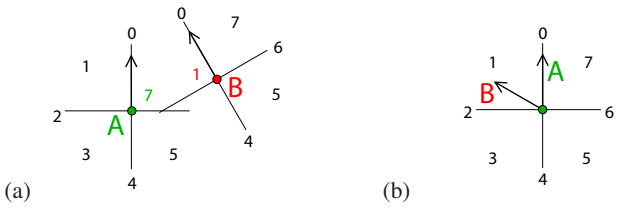



Figure 6.14: $\mathcal{OPRA}_2$ relations between two oriented points: (a) the relation $A\ {}_2\angle{}_7^1\ B$, (b) $A\ {}_2\angle 1\ B$

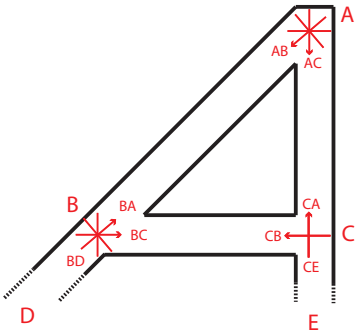
partition the plane into $2m$ planar and $2m$ linear regions. Figure 6.14(a) shows the partition for $\mathcal{OPRA}_2$. The orientation of the two points is depicted by the arrows starting at $A$ and $B$, respectively. The regions are numbered from 0 to $4m-1$. Region 0 always coincides with the orientation of the point. An $\mathcal{OPRA}_m$ base relation $rel_{\mathcal{OPRA}_m}$ is then given by a pair $(i, j)$ where $i$ is the number of the region of $A$ which contains $B$, while $j$ is the number of the region of $B$ which contains $A$. These relations are usually written as $A\ {}_m\angle{}_i^j\ B$, Thus, the example in Fig. 6.14(a) depicts the relation $A\ {}_2\angle{}_7^1\ B$. Additional base relations called *same relations* describe situations in which the positions of both oriented points coincide. In these cases, the relation is determined by the number $s$ of the region of $A$ which contains the orientation arrow of $B$ (as illustrated in Fig. 6.14(b)). These relations are written as $A\ {}_2\angle s\ B$ ($A\ {}_2\angle 1\ B$ in the example). The complete set $\mathcal{R}$ of $\mathcal{OPRA}_m$ relations again is the power set of the base relations.

When we employ the $\mathcal{OPRA}_2$ calculus to describe the relative directions of leaving hallways in the junction observations, the leaving hallways are seen as oriented points positioned on the RGN and pointing in the corresponding direction. The induced spatial description $R(J)$ for a junction observation $J$ then consists of an $\mathcal{OPRA}_2$ relation for each pair of leaving hallways from $L(J)$.

For a relative direction calculus like $\mathcal{OPRA}_2$, only two ways of exploiting the direction information exist, in contrast to the three ways we encountered in the case of absolute direction information. The reason is that enforcement of valid direction ordering is not applicable because no such order exists for relative information. Enforcing valid junction matchings, however, is still possible but now constrains the valid mappings by way of the relations holding between pairs of RGEs.

For the global consistency check, an $\mathcal{OPRA}_2$ constraint network is generated from the route graph hypothesis. Analogously to the generation of the description of a junction observation, one oriented point variable is introduced for each pair of RGN and incident RGE. Hence, we end up with $2 \times n$ variables in the constraint network, where $n$ is the number of RGEs in the hypothesis, while we only had one per junction in the case of absolute cardinal directions. The process of generating the constraint network is illustrated in Fig. 6.15.

The names of the oriented points here are formed from the name of the corres-




| | | |
|---|---|---|
| AB | $_2\angle 1$ | AC |
| BA | $_2\angle 7$ | BC |
| BA | $_2\angle 4$ | BD |
| BC | $_2\angle 5$ | BD |
| CA | $_2\angle 2$ | CB |
| CA | $_2\angle 4$ | CE |
| CB | $_2\angle 2$ | CE |
| AB | $_2\angle^0_0$ | BA |
| AC | $_2\angle^0_0$ | CA |
| BC | $_2\angle^0_0$ | CB |
| BD | $_2\angle^0_0$ | DB |
| CE | $_2\angle^0_0$ | EC |

Figure 6.15: Set of $\mathcal{OPRA}_2$ constraints describing the given route graph hypothesis. A junction of $n$ hallways is represented by $n$ oriented points

ponding RGN and the name of the other RGN incident to the RGE (e.g., AB for the oriented point at RGN A and the RGE leading to B). For each RGN, we generate the constraints holding between each pair of leaving RGEs which are all *same relations*. In addition, we need to state that $XY$ and $YX$ are facing each other (relation $_2\angle^0_0$), forming a single hallway. The complete set of constraints is shown on the right side of the figure.

Finally, consistency again is checked by using the algebraic closure algorithm of SparQ. However, as we already mentioned, this is only an incomplete method that may not discover all inconsistent constraint networks.

### 6.3.3 Incorporation into the Search Algorithm

Planarity checking and spatial direction constraints are incorporated into the search algorithm to discard inconsistent hypotheses as soon as possible and thus prune large subtrees of the search tree.

Spatial direction constraints are utilized in both the update and the expand operations. In the update operation, only valid junction matchings need to be enforced in order to verify that the hypothesis is consistent with the new information. In the expand operation, enforcement of valid direction orderings plays a role when adding a new RGE to an unvisited RGN, but only when an absolute direction calculus is used. Valid junction mappings are enforced when two RGNs are merged. A global consistency check is performed for every successor hypothesis that results from performing a matching.

Planarity checks only need to be performed when a node in the search tree is expanded. The update operation at most appends new RGEs together with a new unvisited RGN and never connects two existing RGNs. However, it still requires that the planarity-related information be updated correctly. In addition, when the match lists of the new RGNs are set, only RGNs that share a face with the new RGN are considered.

In the expand operation, planarity is checked whenever a new edge connecting two existing RGNs is inserted while transforming the graph structure.

The effects of both the planarity constraint and the direction constraints can be seen in the top picture of Fig. 6.8. A significant number of branches have been cut off because inconsistency of the hypothesis has been discovered (marked by the cross). The reasons for discarding a particular node are indicated by the letters below the cross ('p' for planarity, 'd' for direction information). For some nodes, both planarity and spatial consistency are violated. Leaf nodes containing a consistent map hypothesis are tagged by a check mark.

The bottom search tree shows the result of applying pruning based on planarity constraint and direction information together with the branch and bound search. From 112 nodes in the original search tree, only 36 are considered. Ten nodes are rejected because of planarity violation and ten because of inconsistent direction information. While generation of successor nodes and the update operation can be performed in polynomial time (with the global consistency check being the most costly operation), the size of the search tree grows exponentially with the length of history. Therefore, the important question is whether combining constraint-based pruning and best-first search can achieve a sufficient reduction of the search space to make the overall approach feasible. An experimental analysis of this issue based on randomly created graph environments and exploration runs will be conducted in Sect. 7.3.

## 6.4 Combining Minimal Route Graph Mapping and AGVG Representations

In the last section of this chapter, we discuss how the minimal model finding approach developed above can be applied to construct AGVG representations from a sequence of observed Voronoi nodes and traversals of Voronoi curves. Obviously, this global mapping approach should be based on the most relevant of the Voronoi nodes and only add the other nodes when the general topology of the environment has been established.

The AGVG setting deviates in several aspects from the theoretical scenario we studied in the previous sections:

- node signatures provide additional information about RGNs,
- start and end nodes of RGEs (Voronoi curves) can often be perceived together,
- multiple connections between two RGNs are possible,
- reliable perception of direction relations is not given for linear relations or near the sector boundaries,
- connections are typically not straight lines.

The information contained in the signatures of Voronoi nodes can be used as an additional criterion to decide whether two nodes are compatible as described in Chap. 5. The second point in the list refers to the fact that in a Voronoi-based mapping approach the robot typically not only perceives a single Voronoi node but a local Voronoi graph as defined in Sect. 3.5. This kind of information allows us to extend the route graph model without explicitly traversing each edge and, in addition, helps reduce the problems caused by the other deviations from the theoretical framework as we will see below. The simplest way to incorporate this additional information into the framework is by adding virtual traversal actions and junction observations to the history whenever a complete connection is perceived without actually traversing it. These virtual actions and observations simulate traversing the connecting Voronoi curve and then returning to the starting Voronoi node.

Including the possibility of multiple connections between two Voronoi nodes in the framework is straightforward. It only requires a change in the way the match lists are constructed in the update operation. This change, in principle, increases the size of the search space. However, the fact that the adjacent nodes are often part of the local observation means that the difference is negligible in practice.

More serious problems are raised by the last two points in the list. First of all, we cannot expect that the direction relations of leaving Voronoi curves can be completely reliably observed in practice. This is especially true for the linear sectors included in the two direction calculi, which is a general point of criticism with regard to typical qualitative spatial calculi. As a consequence, instead of always employing base relations from the respective calculus, we utilize disjunctions of base relations whenever the perceived direction is a linear relation or lies close to the boundary of a relation sector. For instance, the perceived cardinal direction relation $n$ and a direction belonging to $ne$ but very close to $n$ would both be stored as the disjunction $\{ne, n, nw\}$. When employing disjunctions instead of only base relations, the requirement that directions of matched hallways be identical has to be replaced with the demand that the intersection of the direction relations not be empty.

A further problem for the utilization of direction information is the fact that Voronoi curves are typically not straight connections between two Voronoi nodes but, as the name suggests, curved. Therefore, we cannot expect that a connection leaving node A to the southwest arrives at node B from the northeast and that consequently B has to lie southwest of A. If the connecting Voronoi curve is completely contained in the local observation, this is not a problem as the correct cardinal direction can be read off directly. In situations in which this is not the case, we simply mark the traversal action and refrain from employing the direction information for this edge in the global consistency check. However, we still can use the local direction of the leaving Voronoi curve for matching junction observations.

The last two points and the adaptations made to deal with them mainly concern the pruning of the search space based on global consistency. The extended use of coarse information in the form of disjunctions leads to a diminished inferential power. As a

result the efficiency of this kind of pruning can be significantly reduced. Enforcement of valid direction orderings and valid junction matchings are affected to a lesser degree.

Overall, while these adaptations may sound rather drastic, the effects in practice are less severe because of the already mentioned extended observation range. A quantitative analysis will be performed as part of the evaluation described in the next chapter (Sect. 7.3.5). This analysis will be based on simulated exploration runs through AGVGs of real environments. In addition, we will apply the minimal model approach within an overall Voronoi-based mapping system that combines all the techniques developed in this work in Sect. 7.4.