

# Planning and Optimization

## C4. Delete Relaxation: Relaxed Task Graphs

Malte Helmert and Gabriele Röger

Universität Basel

# Planning and Optimization

## — C4. Delete Relaxation: Relaxed Task Graphs

C4.1 Relaxed Task Graphs

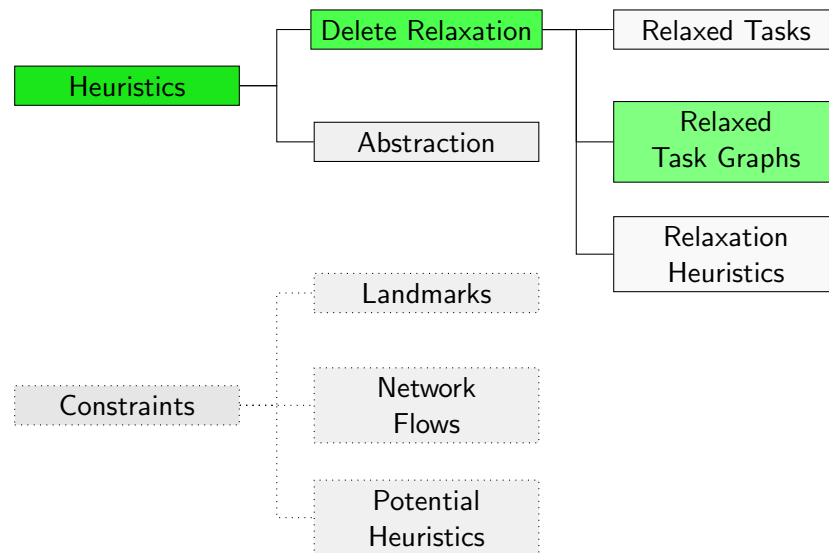
C4.2 Construction

C4.3 Reachability Analysis

C4.4 Remarks

C4.5 Summary

## Content of this Course: Heuristics



# C4.1 Relaxed Task Graphs

## Relaxed Task Graphs

Let  $\Pi^+$  be a relaxed planning task.

The **relaxed task graph** of  $\Pi^+$ , in symbols  $RTG(\Pi^+)$ , is an AND/OR graph that encodes

- ▶ **which state variables** can become true in an applicable operator sequence for  $\Pi^+$ ,
- ▶ **which operators** of  $\Pi^+$  can be included in an applicable operator sequence for  $\Pi^+$ ,
- ▶ if the **goal** of  $\Pi^+$  can be reached,
- ▶ and **how** these things can be achieved.

We present its definition in stages.

**Note:** Throughout this chapter, we assume flat operators.

## Running Example

As a running example, consider the relaxed planning task  $\langle V, I, \{o_1, o_2, o_3, o_4\}, \gamma \rangle$  with

$$V = \{a, b, c, d, e, f, g, h\}$$

$$I = \{a \mapsto \mathbf{T}, b \mapsto \mathbf{T}, c \mapsto \mathbf{F}, d \mapsto \mathbf{T}, e \mapsto \mathbf{F}, f \mapsto \mathbf{F}, g \mapsto \mathbf{F}, h \mapsto \mathbf{F}\}$$

$$o_1 = \langle c \vee (a \wedge b), c \wedge ((c \wedge d) \triangleright e), 1 \rangle$$

$$o_2 = \langle \top, f, 2 \rangle$$

$$o_3 = \langle f, g, 1 \rangle$$

$$o_4 = \langle f, h, 1 \rangle$$

$$\gamma = e \wedge (g \wedge h)$$

## C4.2 Construction

## Components of Relaxed Task Graphs

A relaxed task graph has four kinds of components:

- ▶ **Variable nodes** represent the state variables.
- ▶ The **initial node** represents the initial state.
- ▶ **Operator subgraphs** represent the preconditions and effects of operators.
- ▶ The **goal subgraph** represents the goal.

The idea is to construct the graph in such a way that all nodes representing **reachable** aspects of the task are **forced true**.

## Variable Nodes

Let  $\Pi^+ = \langle V, I, O^+, \gamma \rangle$  be a relaxed planning task.

- ▶ For each  $v \in V$ ,  $RTG(\Pi^+)$  contains an OR node  $n_v$ . These nodes are called **variable nodes**.

## Variable Nodes: Example

$$V = \{a, b, c, d, e, f, g, h\}$$



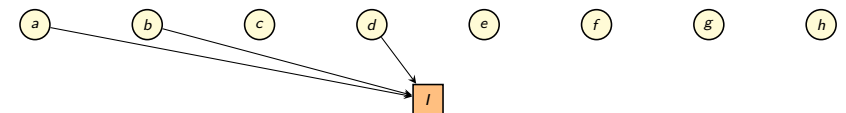
## Initial Node

Let  $\Pi^+ = \langle V, I, O^+, \gamma \rangle$  be a relaxed planning task.

- ▶  $RTG(\Pi^+)$  contains an AND node  $n_I$ . This node is called the **initial node**.
- ▶ For all  $v \in V$  with  $I(v) = \mathbf{T}$ ,  $RTG(\Pi^+)$  has an arc from  $n_v$  to  $n_I$ . These arcs are called **initial state arcs**.
- ▶ The initial node has no successor nodes.

## Initial Node and Initial State Arcs: Example

$$I = \{a \mapsto \mathbf{T}, b \mapsto \mathbf{T}, c \mapsto \mathbf{F}, d \mapsto \mathbf{T}, e \mapsto \mathbf{F}, f \mapsto \mathbf{F}, g \mapsto \mathbf{F}, h \mapsto \mathbf{F}\}$$



## Operator Subgraphs

Let  $\Pi^+ = \langle V, I, O^+, \gamma \rangle$  be a relaxed planning task.

For each operator  $o^+ \in O^+$ ,  $RTG(\Pi^+)$  contains an **operator subgraph** with the following parts:

- ▶ for each formula  $\varphi$  that occurs as a subformula of the precondition or of some effect condition of  $o^+$ , a **formula node**  $n_\varphi$  (details follow)
- ▶ for each conditional effect  $(\chi \triangleright v)$  that occurs in the effect of  $o^+$ , an **effect node**  $n_{o^+}^\chi$  (details follow); unconditional effects are treated as  $(\top \triangleright v)$

## Formula Nodes

Formula nodes  $n_\varphi$  are defined as follows:

- ▶ If  $\varphi = v$  for some state variable  $v$ ,  $n_\varphi$  is the variable node  $n_v$  (so no new node is introduced).
- ▶ If  $\varphi = \top$ ,  $n_\varphi$  is an AND node without outgoing arcs.
- ▶ If  $\varphi = \perp$ ,  $n_\varphi$  is an OR node without outgoing arcs.
- ▶ If  $\varphi = (\varphi_1 \wedge \varphi_2)$ ,  $n_\varphi$  is an AND node with outgoing arcs to  $n_{\varphi_1}$  and  $n_{\varphi_2}$ .
- ▶ If  $\varphi = (\varphi_1 \vee \varphi_2)$ ,  $n_\varphi$  is an OR node with outgoing arcs to  $n_{\varphi_1}$  and  $n_{\varphi_2}$ .

**Note:** identically named nodes are identical, so if the same formula occurs multiple times in the task, the **same** node is reused.

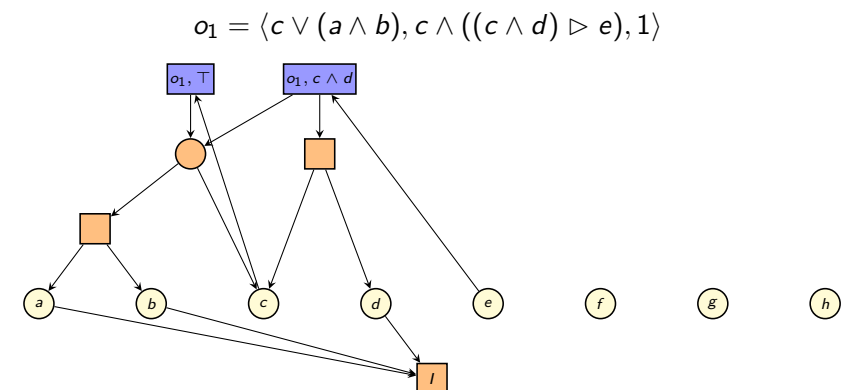
## Effect Nodes

Effect nodes  $n_{o^+}^\chi$  are defined as follows:

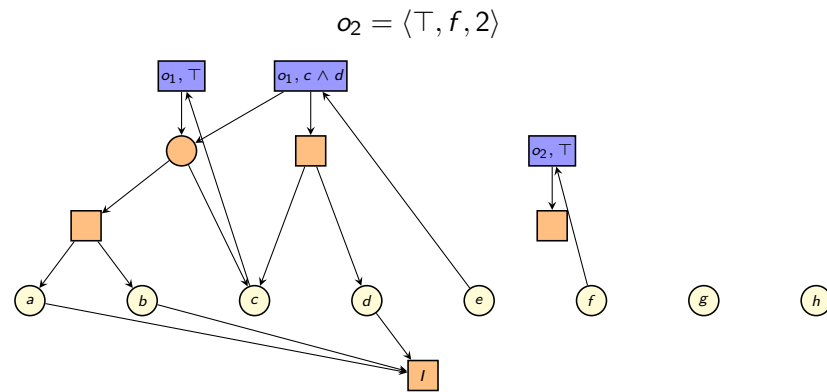
- ▶  $n_{o^+}^\chi$  is an AND node
- ▶ It has an outgoing arc to the formula nodes  $n_{pre(o^+)}$  (**precondition arcs**) and  $n_\chi$  (**effect condition arcs**).
- ▶ Exception: if  $\chi = \top$ , there is no effect condition arc. (This makes our pictures cleaner.)
- ▶ For every conditional effect  $(\chi \triangleright v)$  in the operator, there is an arc from variable node  $n_v$  to  $n_{o^+}^\chi$  (**effect arcs**).

**Note:** identically named nodes are identical, so if the same effect condition occurs multiple times in the same operator, this only induces **one** node.

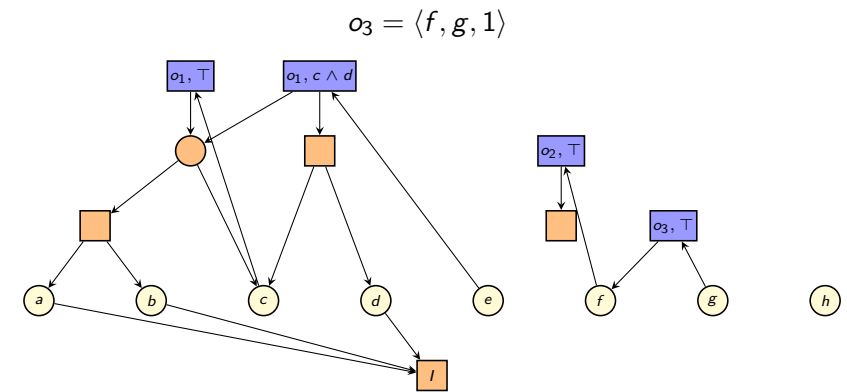
## Operator Subgraphs: Example



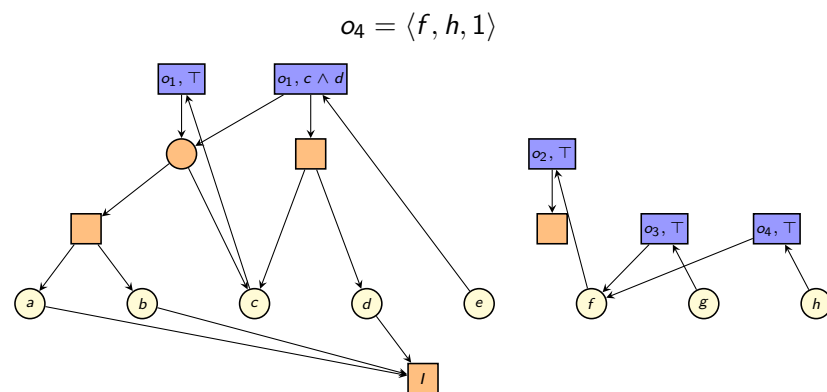
## Operator Subgraphs: Example



## Operator Subgraphs: Example



## Operator Subgraphs: Example

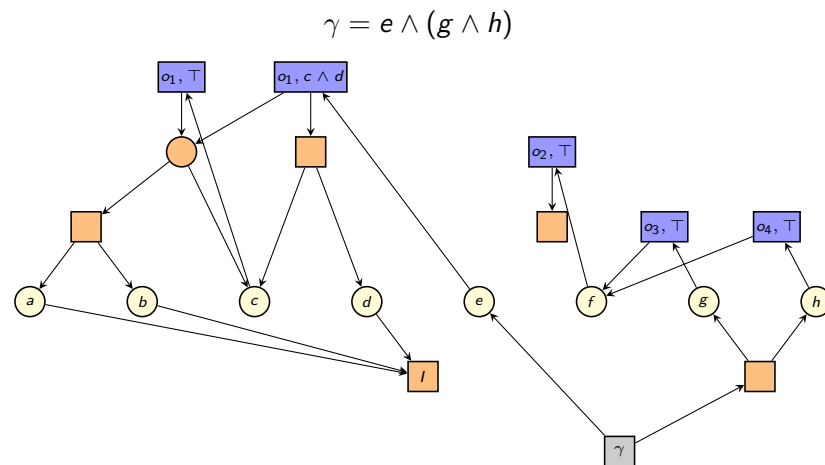


## Goal Subgraph

Let  $\Pi^+ = \langle V, I, O^+, \gamma \rangle$  be a relaxed planning task.

$RTG(\Pi^+)$  contains a **goal subgraph**, consisting of formula nodes for the goal  $\gamma$  and its subformulas, constructed in the same way as formula nodes for preconditions and effect conditions.

## Goal Subgraph and Final Relaxed Task Graph: Example



## C4.3 Reachability Analysis

## How Can We Use Relaxed Task Graphs?

- ▶ We are now done with the definition of relaxed task graphs.
- ▶ Now we want to **use** them to derive information about planning tasks.
- ▶ In the following chapter, we will use them to compute heuristics for delete-relaxed planning tasks.
- ▶ Here, we start with something simpler: **reachability analysis**.

## Forced True Nodes and Reachability

### Theorem (Forced True Nodes vs. Reachability)

Let  $\Pi^+ = \langle V, I, O^+, \gamma \rangle$  be a relaxed planning task, and let  $N_{\mathbf{T}}$  be the forced true nodes of  $RTG(\Pi^+)$ .

For all **formulas** over state variables  $\varphi$  that occur in the definition of  $\Pi^+$ :

**$\varphi$  is true in some reachable state of  $\Pi^+$  iff  $n_{\varphi} \in N_{\mathbf{T}}$ .**

(We omit the proof.)



## Relaxed Task Graphs in the Literature

Some remarks on the planning literature:

- ▶ Usually, only the **STRIPS** case is studied.
- ↔ definitions simpler: only **variable nodes** and **operator nodes**, no formula nodes or effect nodes
- ▶ Usually, so-called **relaxed planning graphs** (RPGs) are studied instead of RTGs.
- ▶ These are **temporally unrolled** versions of RTGs, i.e., they have multiple layers (“time steps”) and are acyclic.
- ↔ Foundations of Artificial Intelligence course FS 2020, Ch. 35–36

## C4.5 Summary

## Summary

- ▶ **Relaxed task graphs** (RTGs) represent (most of) the information of a relaxed planning task as an AND/OR graph.
- ▶ They consist of:
  - ▶ **variable nodes**
  - ▶ **an initial node**
  - ▶ **operator subgraphs** including **formula nodes** and **effect nodes**
  - ▶ a **goal subgraph** including **formula nodes**
- ▶ RTGs can be used to analyze **reachability** in relaxed tasks: forced true nodes mean “reachable”, other nodes mean “unreachable”.