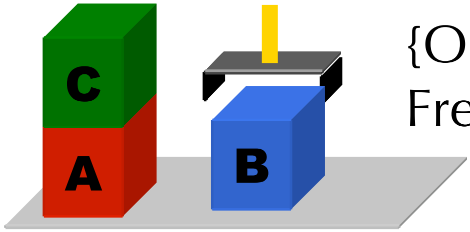
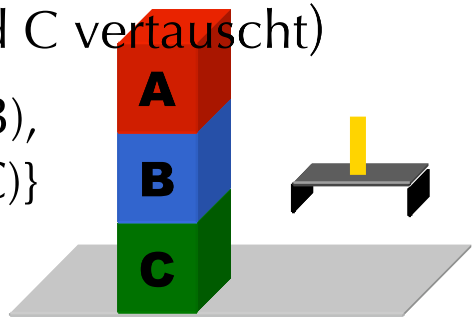
Teilzielinteraktion

Ein Grund für die NP-Vollständigkeit des propositionalen Planens:
Ziele **interagieren**, sind **nicht serialisierbar** (nur *nearly decomposable*)

Beispiel („**Sussmans Anomalie**", wie vorher, nur A und C vertauscht)



{Ontbl(A), Ontbl(B), Free(B), Free(C), On(C,A), Hand(NIL)} → {On(A,B), On(B,C)}



- Zielwahl in 2.1 **erst On(B,C)**: Plan startet mit **Pick**(B), **Stack**(B,C), muss danach wieder entfernt werden
- Zielwahl in 2.1 **erst On(A,B)**: Plan setzt A auf B, muss dann wieder entfernt werden
- Optimaler STRIPS-Plan (mit Alternative 2):
  ⟨**Unstack**(C,A), **Put**(C), **Pick**(A), **Stack**(A,B), **Unstack**(A,B), **Put**(A), **Pick**(B), **Stack**(B,C), **Pick**(A), **Stack**(A,B)⟩

Wohlgeformtheit

Planraumsuche erlaubt flexiblere Suche und variablere Pläne als mit der STRIPS-Funktion erreichbar.
Dazu zuvor einige Definitionen.

Erzeuger, Nutzer, Zerstörer als Merkmale von Operatoren

Stack(x,y)

Vorbedingung	Effekte
Hand(x), Free(y)	On(x,y), Free(x), Hand(NIL)
	–Hand(x), –Free(y)

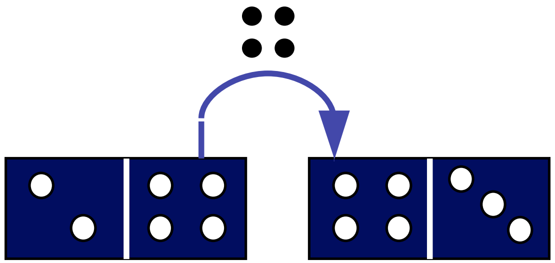
Nutzer

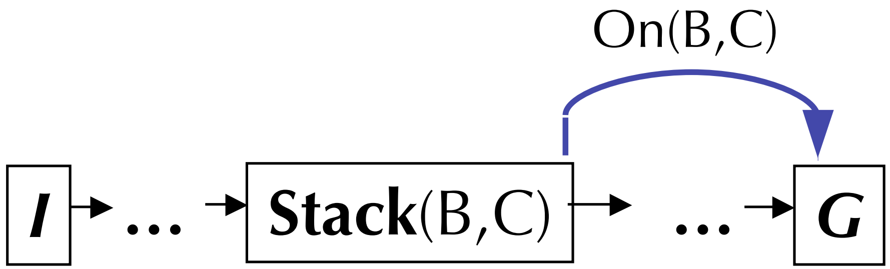
Erzeuger

Zerstörer

Ein Plan $\langle O, <\rangle$ ist **wohlgeformt**, wenn für alle Fakten f und alle $n \in O$, die f nutzen, ein f-Erzeuger $e \in O$ existiert, sodass $e < n$.

Abhängigkeiten





$n \in O$ ist **abhängig** von $e \in O$ bzgl. f in $\langle O, < \rangle$ gdw.

- e erzeugt f; n nutzt f, $e<n$;
- kein $o \in O$ produziert f, sodass $e<o<n$

Im Englischen: *dependencies, causal links*

Konflikte

Die Planordnung muss nach Definition nicht vollständig sein!
(*least commitment*-Strategie)

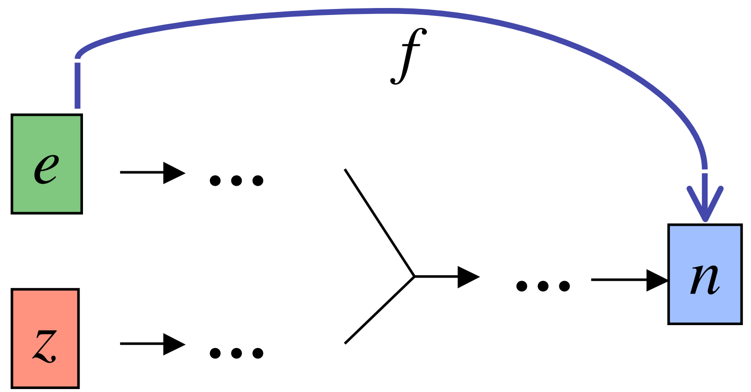
Interpretation der Ungeordnetheit von Operatoren o,p:

o,p können in beliebiger Reihenfolge ausgeführt werden

Es gibt *keine* Parallelausführung in klassischen Plänen!

$\langle O, <\rangle$ enthält einen **Konflikt**, zwischen $z \in O$ und einer Abhängigkeit bzgl. e,f,n gdw.

- z zerstört f;
- es gibt eine Erweiterung $<'$ der Ordnung $<$, sodass $e <' z <' n$ und für alle $o \in O$ gilt: Wenn $z <' o <' n$, so ist o kein f-Produzent



Die Prozedur POP$\langle O,<\rangle$

1. **if** the ordering relation $<$ contains a cycle, **then return fail**
2. **if** $\langle O, <\rangle$ is well-formed and conflict free, **then return** $\langle O, <\rangle$
3. **choose** an operator $o \in O$ with an unresolved precondition m;
4. **if** m is unresolved due to conflict between z and a dependency wrt. e,m,o
5. **then** **choose** one of the alternatives
   - **Promotion**: $< \leftarrow < \cup \{(z,e)\}$
   - **Demotion**: $< \leftarrow < \cup \{(o,z)\}$
   - **Sort in**: **choose** a m-producer $e' \in O$, which is un-ordered wrt. z, such that $e \neq e'$; $< \leftarrow < \cup \{(z,e'), (e',o)\}$;
   - **Insert**: **choose** a new m-producer $e' \notin O$; $< \leftarrow < \cup \{(z,e'), (e',o)\}$
6. **else** **choose** an m-producer e' according to one of
   - **Sort in**: **choose** $e' \in O$, $\neg[e' < o]$; $< \leftarrow < \cup \{(e',o)\}$
   - **Insert**: **choose** a new m-producer $e' \notin O$; $O \leftarrow O \cup \{e'\}$; $< \leftarrow < \cup \{(\boldsymbol{I},e'),(e',o)\}$
7. **return** POP$\langle O,<\rangle$

Beispielproblem: Platten Reifen wechseln

aus: Russell/Norvig, deutsche Ausgabe

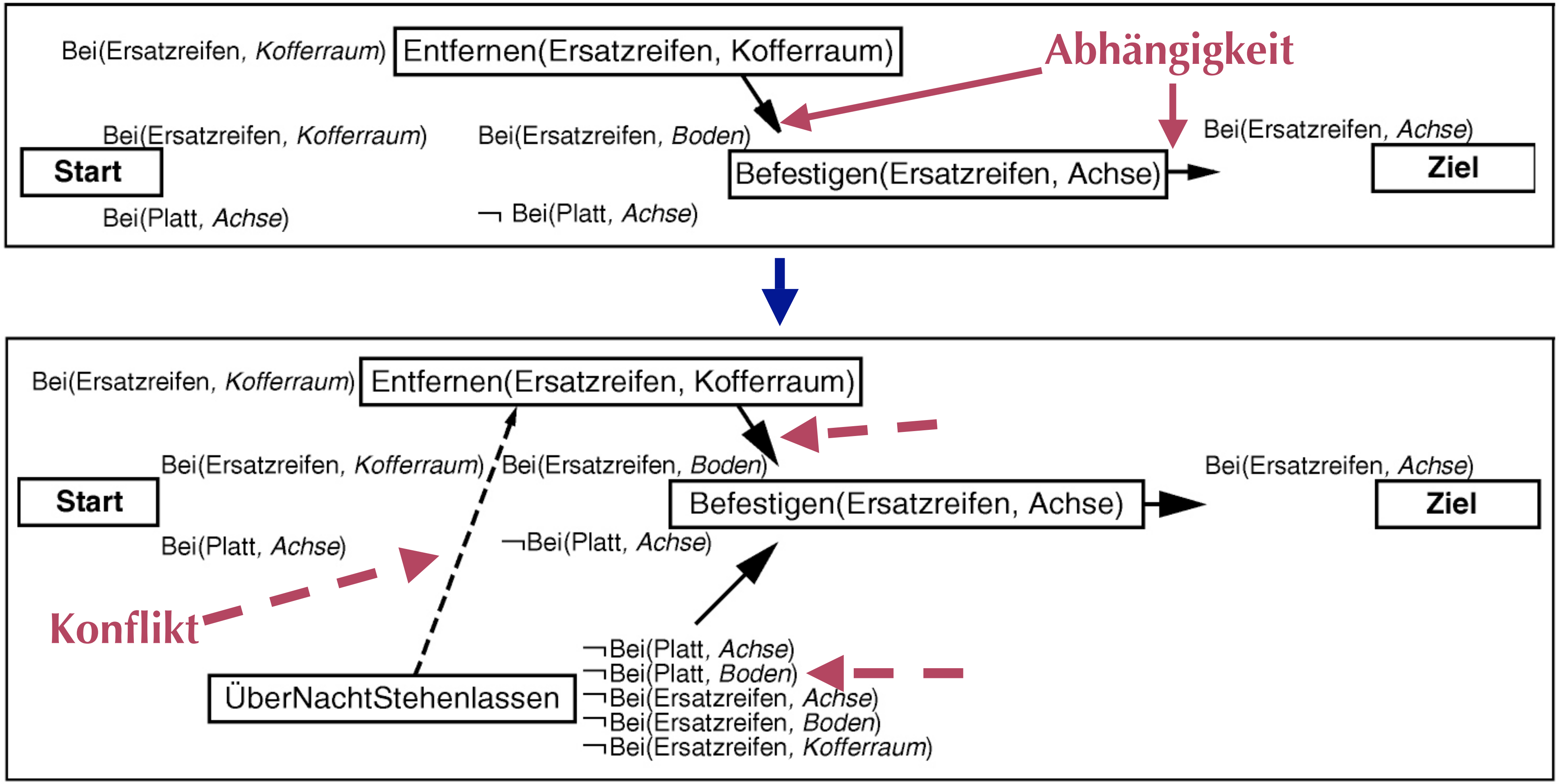
Start

Initiieren(*Bei*(*Platt*, *Achse*) ∧ *Bei*(*Ersatzreifen*, *Kofferraum*))
Ziel(~~*Bei*(*Platt*, *Achse*)~~) *Bei(Ersatzreifen, Achse)*
Aktion(*Entfernen*(*Ersatzreifen*, *Kofferraum*),
  PRECOND: *Bei*(*Ersatzreifen*, *Kofferraum*)
  EFFECT: ¬ *Bei*(*Ersatzreifen*, *Kofferraum*) ∧ *Bei*(*Ersatzreifen*, *Boden*))
Aktion(*Entfernen*(*Platt*, *Achse*),
  PRECOND: *Bei*(*Platt*, *Achse*)
  EFFECT: ¬ *Bei*(*Platt*, *Achse*) ∧ *Bei*(*Platt*, *Boden*))
Aktion(*Befestigen*(*Ersatzreifen*, *Achse*),
  PRECOND: *Bei*(*Ersatzreifen*, *Boden*) ∧ ¬ *Bei*(*Platt*, *Achse*)
  EFFECT: ¬ *Bei*(*Ersatzreifen*, *Boden*) ∧ *Bei*(*Ersatzreifen*, *Achse*))
Aktion(*ÜberNachtStehenlassen*),
  PRECOND:
  EFFECT: ¬ *Bei*(*Ersatzreifen*, *Boden*) ∧ ¬ *Bei*(*Ersatzreifen*, *Achse*)
    ∧ ¬ *Bei*(*Ersatzreifen*, *Kofferraum*) ∧ ¬ *Bei*(*Platt*, *Boden*)
    ∧ ¬ *Bei*(*Platt*, *Achse*))

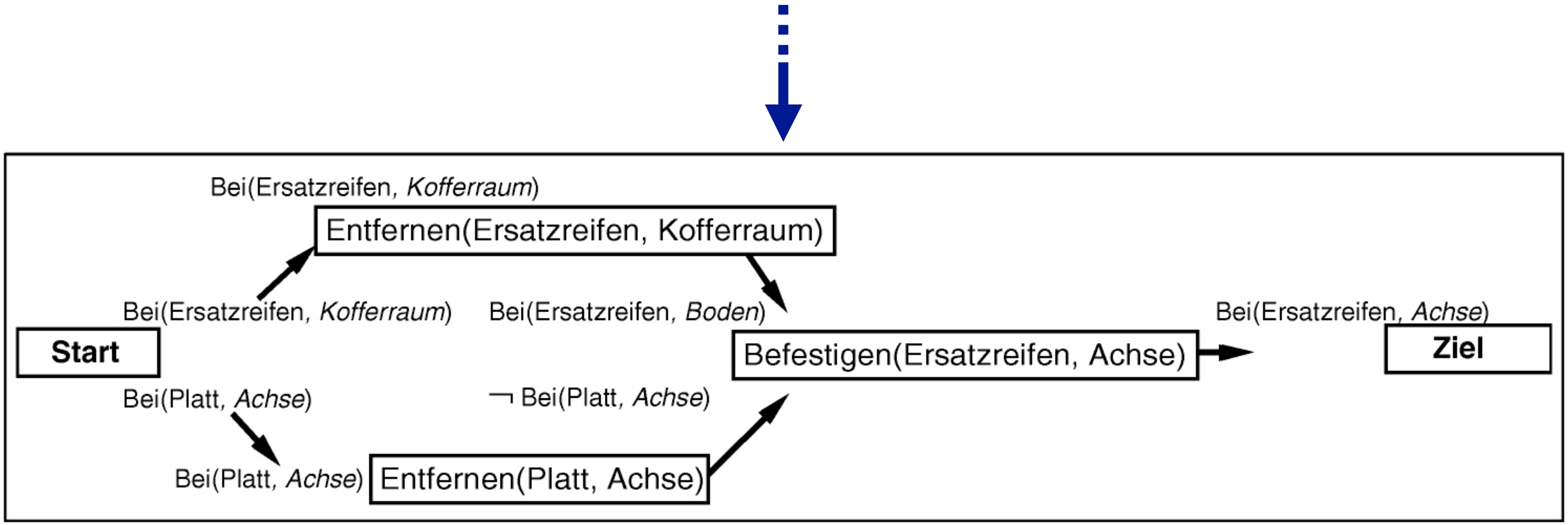
Aktionsinstanzen

Mögliche POP-Zustände im Reifenproblem

- **Start** ≡ ***I***; **Ziel** ≡ *G*
- Ordnungskanten, Op.-Nachbedingungen weggelassen



Lösungsplan fürs Reifenproblem



- Planordnung ist *nichtlinear*:
  Entferne Ersatzreifen und platten Reifen in beliebiger Reihenfolge
- Da POP keine Operatoren im Plan löschen kann, kommt man nur durch Backtracking vom vorigen Plan zu dieser Lösung!

Heuristiken für POP-Planungskosten

- zulässige Heuristiken problematisch
  - # offene (*unresolved*) Ziele i.A. *unzulässig*, aber trotzdem oft allzu weit *unterschätzend*
  - A*-artige Suche meist zu aufwändig
- *most constrained choice*
- Bereichsspezifische Heuristiken
  - z.B. Blockwelt: „Setze alle Blöcke auf den Tisch, die nicht in Zielen erwähnt; dann baue Türme von unten“

HTN-Planen

HTN = Hierarchical Transition Network
Verwendung von „Unter-Plänen“ beim Planen

- Spart Planungszeit durch Verwendung von „Makros“
- Erlaubt im Domänenmodell „Vorwissen“ zu formulieren
- Vereinfacht Plan-Darstellung für Benutzer
- Praktisch alle Planungs-Anwendungssysteme enthalten HTNs

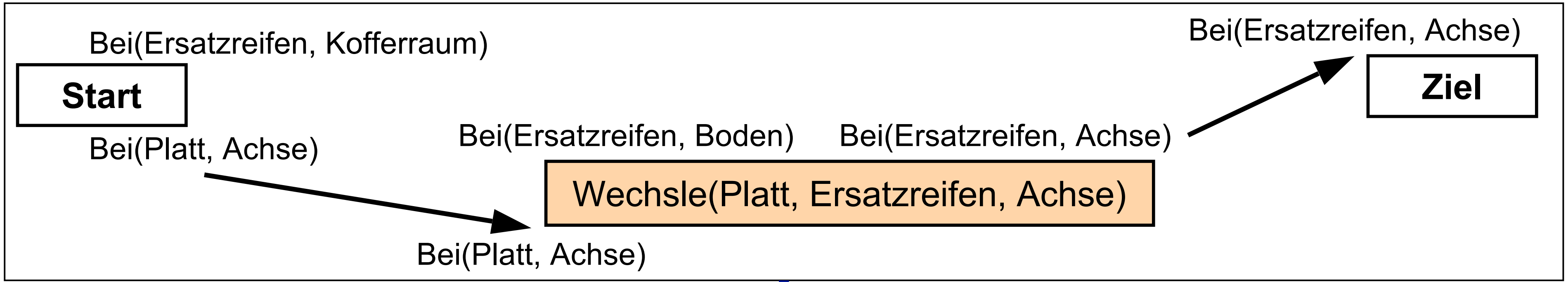
Erweitern von POP mit HTNs

Erweiterung der Reifendomäne durch einen H-Operator

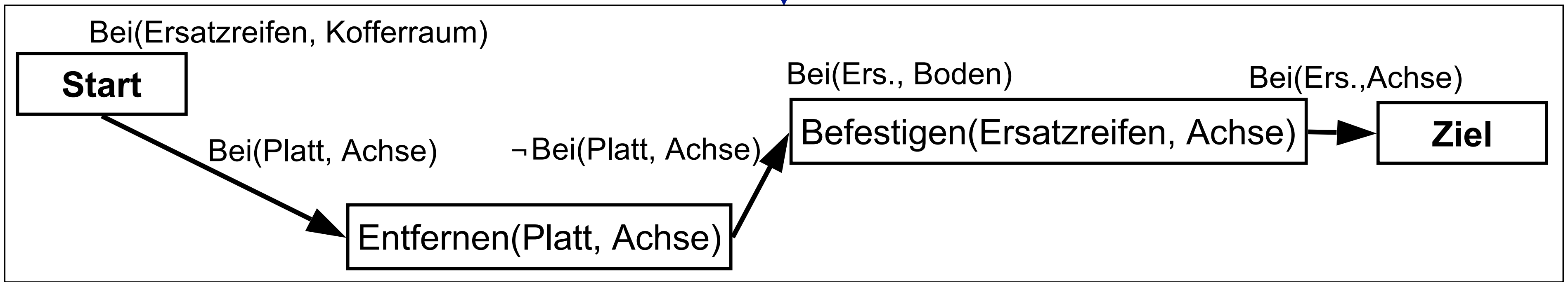
H-Aktion(*Wechsle*(*Platt*,*Ersatzreifen*,*Achse*),
PRECOND: *Bei*(*Platt*,*Achse*) ∧ *Bei*(*Ersatzreifen*,*Boden*)
EFFEKT: *Bei*(*Ersatzreifen*,*Achse*)
EXPAND: OPS: {S1: *Entfernen*(*Platt*,*Achse*), S2: *Befestigen*(*Ersatzreifen*,*Achse*)}
ORD: {S1 < S2}
ABH: {⟨S1, ¬*Bei*(*Platt*,*Achse*), S2⟩}

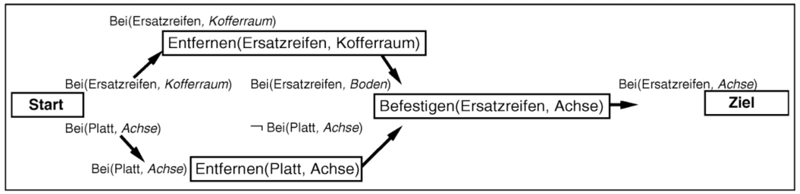
- H-Operatoren können wie Operatoren eingesetzt werden, um ungelöstes Merkmal zu erzeugen
  - Heuristik: Verwende H-Operatoren mit Priorität
- Sorge bei Expansion für richtige „Vererbung“ der Abhängigkeiten
- Vor Terminierung müssen alle H-Operatoren expandiert sein
  - Heuristik: Expandiere früh

Beispiel: HTN-POP



Expansion



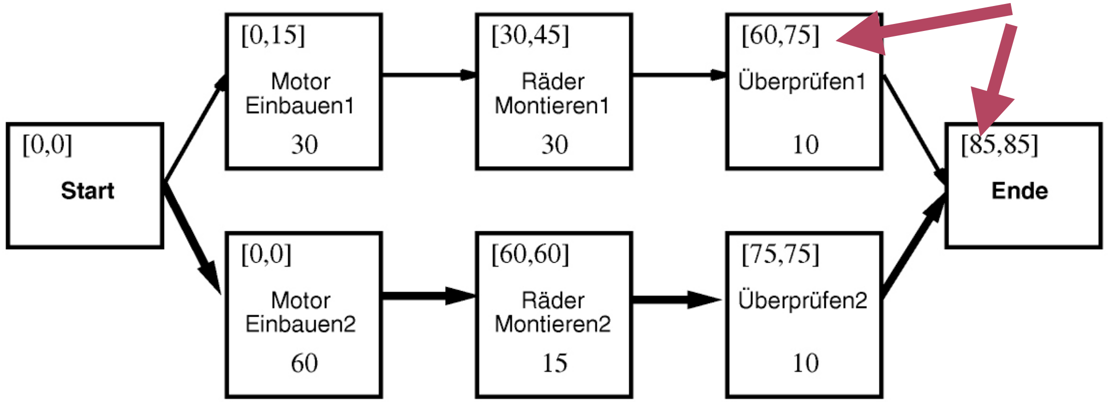


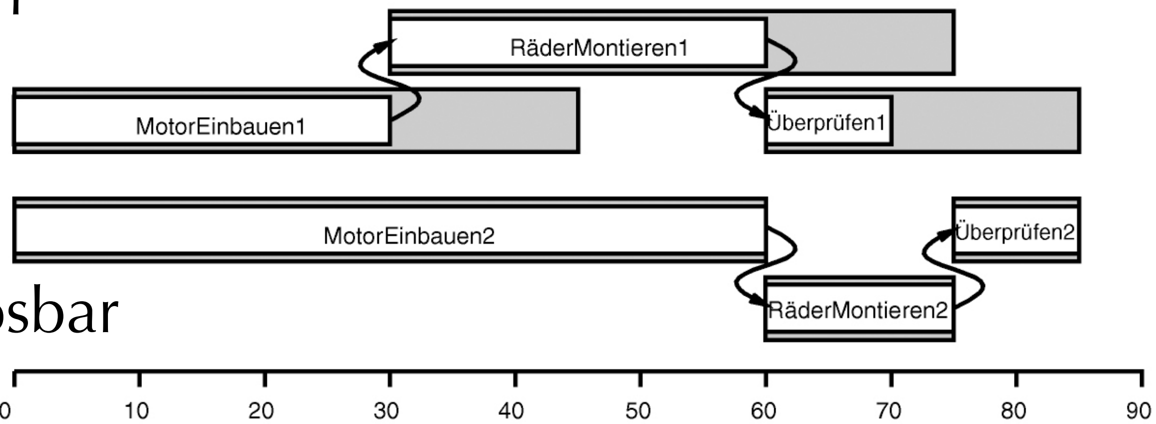
Lösungsplan
wie zuvor

Termin/Zeitplanung (*Scheduling*)

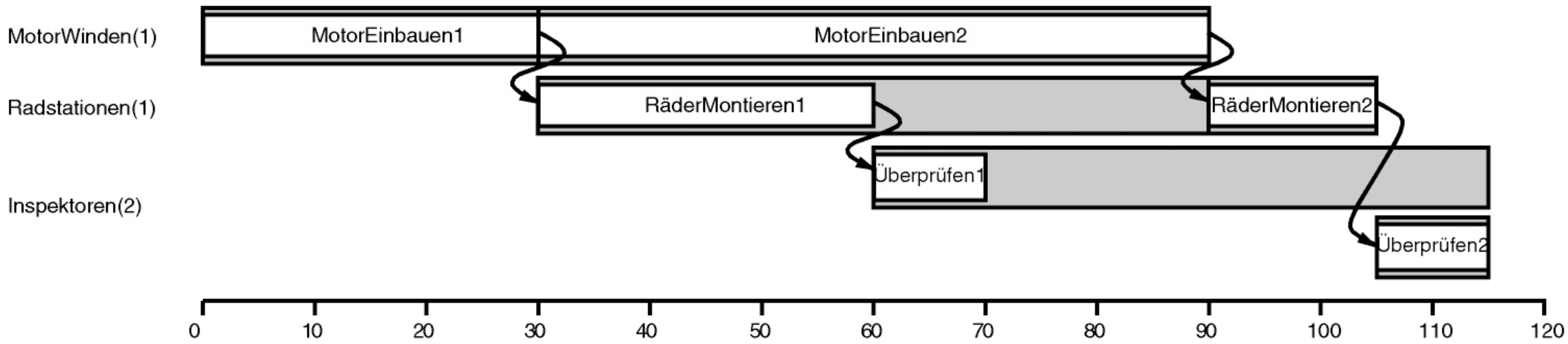
- Klassisches Thema im ***Operations Research*** („Netzplantechnik“)
  - **Fertigungsplanung** (*job shop scheduling*): Gegeben Aufträge mit Terminen, Maschinen und Fertigungsreihenfolgen (= partiell geordnete Pläne!), terminiere (Teil-)Aufträge
- In einfachster Variante lösbar über **Kritische Pfade** (polynomiell)

s. **STNs**





Terminplanung bei Ressourcenbegrenzung



- Ebenfalls klassisches Thema im OR
- NP-vollständig
- Großes Potenzial durch Kombination von Planung und ressourcensensitiver Terminierung (Logistik, Fertigung)

"The deployment of a single logistics support aid called DART during the Desert Shield/ Desert Storm Campaign paid back all US government investment on AI/KBS research over a 30 year period."
[*Critical Technology Assessment of the U.S. Artificial Intelligence Sector*, U.S.Dept. of Commerce, 1994]

Effizienzgewinn

- Klassisches Planen ist oft propositionales Planen, Algorithmen und Repräsentationen sind aber erweiterbar (Variable, Fkt.en)
- POP akzeptiert einige „Dogmen“, die letztlich in dieser Erweiterbarkeit begründet sind
  - Rückwärtssuche, *Systematizität* (keine Operatoren löschen)
- Seit den späten 1990ern werden Algorithmen entwickelt, die konsequent nur für propositionales Planen gedacht sind und dessen Randbedingungen ausnutzen (z.B. GRAPHPLAN, 1996)
- Zusammen mit „alter“ POP-Erfahrung haben sie in wenigen Jahren zu Effizienzgewinn in Größenordnungen geführt
- POP ist derzeit außer Mode
- Propositionales Planen **kann aber nicht das letzte Wort sein!**

Planungsgraphen: Vorbereitung

Schlüsselidee vieler moderner propositionaler Planer:

- Kombiniere *Planungsheuristik* (Ignorieren negativer Nachbedingungen) mit *Vorwärtssuche* zum Aufspannen des Planraums in kompakter Darstellung (**Planungsgraph**)
- Verwende Planungsgraph als notwendige Bedingung für Existenz eines Plans der „Tiefe“ d
- Extrahiere Plan ggf. aus dem Planungsgraphen

Sei f ein Grund-Fakt (Aussagevariable oder Prädikat mit grundinstanzierten Argumenten) einer Domänenbeschreibung. Der **Persistenzoperator** **Per**(f) ist ein Pseudo-Operator mit der Vorbedingung f und der Nachbedingung f.

Grafisch: $^{f}\boxed{\mathbf{Per}(f)}^{f}$ oder $f\,\square\,f$

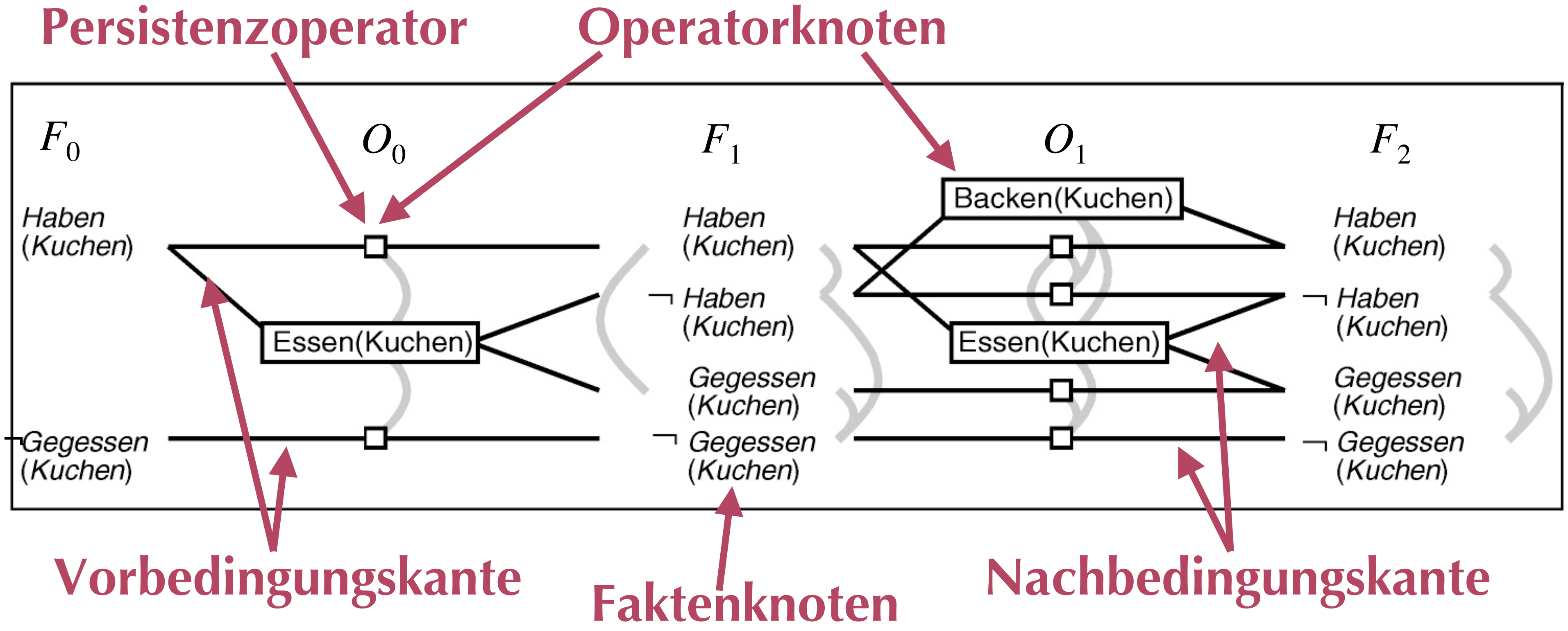
Planungsgraph

Seien $\langle S,O,Z\rangle$ ein propositionales Planungsproblem in einer Domäne D, O' die Persistenzoperatoren in D.
Ein **Planungsgraph** $G_n=\langle N,E\rangle$ der Ordnung n für $\langle S,O,Z\rangle$ ist ein gerichteter, bipartiter Schichtgraph, wobei gilt:

- $N = [F_0;\ O_0;\ F_1;\ O_1;\ \ldots;\ O_{n-1};\ F_n]$; die Knoten in $O_i \subseteq O \cup O'$ heißen **Operatorknoten**; die Knoten in F_i entsprechen Grund-Fakten und heißen **Faktenknoten**.
- $F_0=S$; $\quad \forall i>0$: $[f \in F_i \Leftrightarrow \exists \langle V,N\rangle \in O_{i-1}:[f \in N]]$
- $\forall i$: $\forall \langle V,N\rangle = o \in O \cup O'$: $[V \subseteq F_i \Leftrightarrow o \in O_i]$
- $E = E_V \cup F_N$
- $E_V \subseteq F_i \times O_i$, $i=\{0,\ldots,n-1\}$, heißen **Vorbedingungskanten** und verbinden Faktenknoten, die den Vorbedingungen eines Operators o entsprechen, mit dem Operatorknoten $o \in O_i$.
- $E_N \subseteq O_{i-1} \times F_i$, $i=\{1,\ldots,n\}$, heißen **Nachbedingungskanten**. Sie verbinden Operatorknoten mit Faktenknoten, die den Nachbedingungen entsprechen.

Beispiel: *Have the Cake and Eat it, too*

graue, gebogene Kanten zunächst ignorieren!



- Merkmale akkumulieren über die F_i
- Merkmale in einzelnen F_i „widersprechen“ sich ↪ graue Kanten

Mutex: Wechselseitiger Ausschluss

Mutex für Englisch *mutual exclusion*

Sei $G_n=\langle N,E\rangle$ ein Planungsgraph der Ordnung n für $\langle S,O,Z\rangle$.

- Zwei Operatoren **interferieren**, gdw. der eine eine Vorbedingung oder einen Effekt des anderen löscht.
- Zwei Operatoren **konkurrieren** auf Schicht O_i, gdw. sie Fakten unter ihren Vorbedingungen haben, die sich auf der Vorgängerschicht F_i ausschließen.
- Zwei Operatoren innerhalb einer Schicht **schließen sich aus**,
  - in Schicht O_0, gdw. sie interferieren
  - in Schicht $O_{i>1}$, gdw. sie interferieren oder konkurrieren
- Zwei Fakten f,g **schließen sich** in Schicht $F_{i>0}$ **aus**, gdw. es keine oder nur wechselseitig ausschließende Operatoren in der Vorgängerschicht O_{i-1} gibt, die f und g erzeugen.

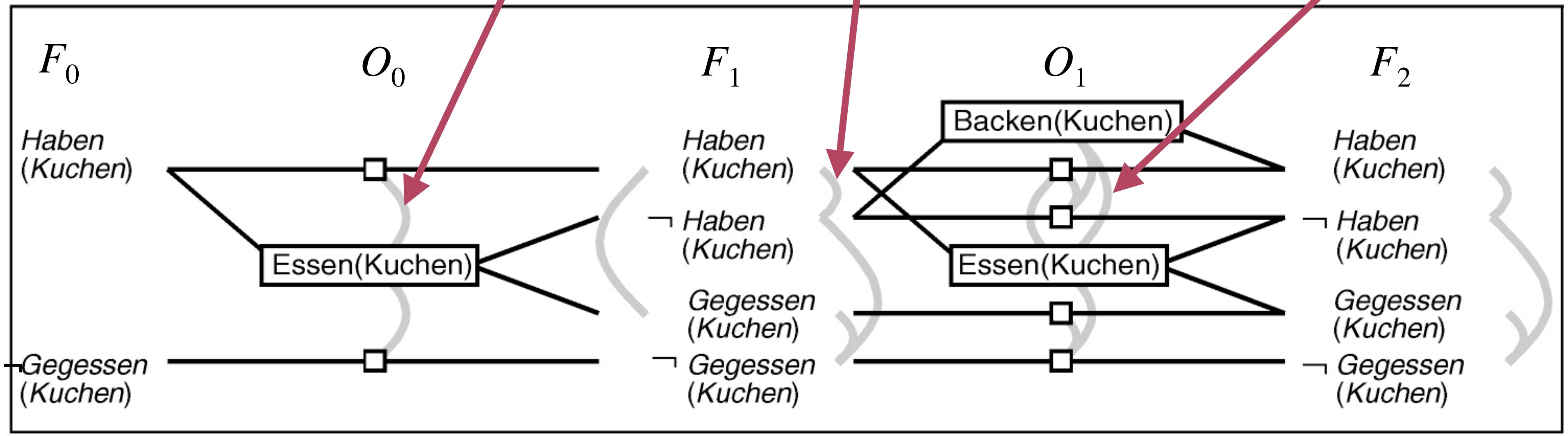
Mutex beim Kuchenessen

graue, gebogene Kanten: Mutex (nicht alle eingezeichnet)

Interferenz
±Haben(Kuchen)

Ausschluss
erzeugt durch
Mutex-Op.en

Konkurrenz
Haben(Kuchen)
vs.
¬Haben(Kuchen)



Idee für Algorithmus: Extrahiere Mutex-freien Lösungsplan!