AI Planning

5. Causal Graphs

How to Capture Problem *Structure*

Álvaro Torralba, Cosmina Croitoru

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Winter Term 2018/2019

Thanks to Prof. Jörg Hoffmann for slide sources

Agenda

Motivation

What is general problem solving all about?

→ Writing a program that is not specialized to a particular problem.

How can such a program be effective?

→ By self-adapting to any particular problem it is given as input.

What does the program require, to be able to self-adapt?

→ Understand and exploit the structure of the problem.

But then, what should we – as researchers – do, first of all?

→ Figure out what "problem structure" actually is.

→ Causal graphs capture the structure of the planning task input, in terms of the direct **dependencies between state variables**. This can be exploited for many different purposes.

Example Uses of Causal Graphs

- **Chapter 10:** Identifying a sub-class of planning tasks where red-black planning (generating a partial delete relexation heuristic) is tractable.
- **Chapter 12:** Avoiding redundant work in the search for a pattern collection when generating a pattern database heuristic.
- Search space surface analysis. Identifying a sub-class of planning tasks where h^+ provably has no local minima [Hoffmann (2011)].
- Complexity analysis: [Domshlak and Dinitz (2001); Brafman and Domshlak (2003); Katz and Domshlak (2008); Giménez and Jonsson (2009); Chen and Giménez (2010); Katz and Keyder (2012)].
- Designing and generating (yet more) heuristic functions: [Helmert (2004); Katz and Domshlak (2010); Domshlak *et al.* (2015)].
- System design: Guaranteeing desired behavior [Williams and Nayak (1997)].
- Factorized planning: Problem decomposition [Knoblock (1994); Brafman and Domshlak (2013); Gnad and Hoffmann (2015)].

Our Agenda for This Chapter

2. **Causal Graphs:** For explicitly capturing the "internal structure" of a planning task, causal graphs are by far the most prominent notion in the planning literature.
3. **Domain Transition Graphs:** These are simple graphs describing the behavior of individual state variables; they are often considered in connection with causal graphs.
4. **Example Results:** We show some examples of causal graph based analyses, which are easy to explain and will be useful later on in **Chapters 10 and 12**).

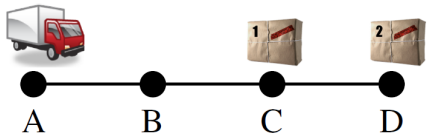
Causal Graphs

Definition (Causal Graph). *Let $\Pi = (V, A, c, I, G)$ be an FDR planning task. The causal graph of Π is the directed graph $CG(\Pi)$ with vertices V and an arc (u, v) if $u \neq v$ and there exists an action $a \in A$ so that either (i) there exists $a \in A$ so that $pre_a(u)$ and $eff_a(v)$ are both defined, or (ii) there exists $a \in A$ so that $eff_a(u)$ and $eff_a(v)$ are both defined.*

Causal graphs capture variable dependencies:

- Arc (i) (u, v): we may have to move u in order to be able to move v.
- Arc (ii) (u, v): moving u may, as a side effect, move v as well.
  $\rightarrow$ Note that we always also get the arc (v, u) in this situation, constituting a cycle between u and v.

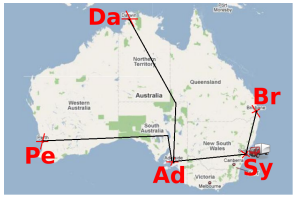
Example: "Logistics"



- State variables V: $truck : \{A, B, C, D\}$; $pack1, pack2 : \{A, B, C, D, T\}$.
- Initial state I: $truck = A$, $pack1 = C$, $pack2 = D$.
- Goal G: $truck = A$, $pack1 = D$.
- Actions A (unit costs): $drive(x, y)$, $load(p, x)$, $unload(p, x)$. E.g.: $load(pack1, x)$ precondition $truck = x, pack1 = x$, effect $pack1 = T$.

Causal graph?

Example: "TSP"



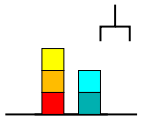
- Variables V: $at : \{Sy, Ad, Br, Pe, Da\}$; $v(x) : \{T, F\}$ for $x \in \{Sy, Ad, Br, Pe, Da\}$.
- Initial state I: $at = Sy, v(Sy) = T, v(x) = F$ for $x \neq Sy$.
- Goal G: $at = Sy, v(x) = T$ for all x.
- Actions A: $drive(x, y)$ where x, y have a road; pre $at = x$, eff $at = y, v(y)$.
- Cost function c: $Sy \leftrightarrow Br : 1$, $Sy \leftrightarrow Ad : 1.5$, $Ad \leftrightarrow Pe : 3.5$, $Ad \leftrightarrow Da : 4$.

Causal graph?

Causal Graphs Cycles: Class (ii) Effect-Effect

Abstract example: If $V = \{u, v\}$ and $A = \{a\}$ with $\mathit{eff}_a = \{u = d, v = e\}$, the causal graph has arcs (u, v) and (v, u).

Less abstract example: Blocksworld.

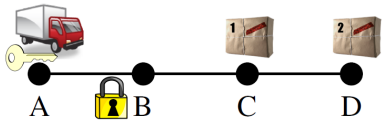


- n blocks, 1 hand.
- A single action either takes a block with the hand or puts a block we're holding onto some other block/the table.
- For example, say $\mathit{pickup}(x, y)$ has precondition $\mathit{atx} = y$, $\mathit{clear}x = \mathit{true}$, $\mathit{handEmpty} = \mathit{true}$; and effect $\mathit{atx} = \mathit{hand}$, $\mathit{cleary} = \mathit{true}$, $\mathit{handEmpty} = \mathit{false}$.

→ So there are class (ii) cycles in the Blocksworld, for example between variables of the form "atx" and "cleary".

→ Class (ii) (effect-effect) causal graph cycles occur whenever an action has more than one effect. Their absence is equivalent to "unary" actions, each affecting only a single variable.

Causal Graphs Cycles: Class (i) Precondition-Effect



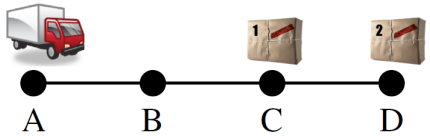
- State variables V: truck, $\mathit{pack1}$, $\mathit{pack2}$ as before; $\mathit{atKey} : \{A, B, C, D, T\}$.
- Initial state I: $\mathit{truck} = A$, $\mathit{pack1} = C$, $\mathit{pack2} = D$, $\mathit{atKey} = A$.
- Goal G: $\mathit{truck} = A$, $\mathit{pack1} = D$.
- Actions A: As before; and $\mathit{takeKey}(x)$ with pre $\mathit{truck} = x, \mathit{atKey} = x$, effect $\mathit{atKey} = T$; and $\mathit{drive}(A, B)$ has additional pre $\mathit{atKey} = T$.

→ Are there class (i) cycles in this example?

→ Class (i) (precondition-effect) causal graph cycles occur when there are "cyclic support dependencies", where moving variable x requires a precondition on y which (transitively) requires a precondition on x.

Where Causal Graphs Fail

Why not in STRIPS?



- Facts P: $\mathit{truck}(x)\, x \in \{A, B, C, D\}$; $\mathit{pack1}(x), \mathit{pack2}(x)\, x \in \{A, B, C, D, T\}$.
- Initial state I: $\{\mathit{truck}(A), \mathit{pack1}(C), \mathit{pack2}(D)\}$.
- goal G: $\{\mathit{truck}(A), \mathit{pack1}(D)\}$.
- Actions A (unit costs): $\mathit{drive}(x, y)$, $\mathit{load}(p, x)$, $\mathit{unload}(p, x)$. E.g.: $\mathit{load}(\mathit{pack1}, x)$ pre $\mathit{truck}(x), \mathit{pack1}(x)$, add $\mathit{pack1}(T)$, del $\mathit{pack1}(x)$.

Causal graph?

→ Reminder **Chapter 2**: "Causal graphs have a much clearer structure for FDR (e.g., acyclic vs. cyclic)."

Questionnaire

Question!

In which of these domains does the causal graph of an FDR encoding contain cycles?

(A): Finding a path in a graph
(B): Sokoban
(C): Logistics with fuel consumption
(D): Logistics with many trucks and packages

Domain Transition Graphs

Definition (Domain Transition Graph). *Let $\Pi = (V, A, c, I, G)$ be an FDR planning task, and let $v \in V$. The domain transition graph (DTG) of v is the labeled directed graph $DTG(v, \Pi)$ with vertices D_v and an arc (d, d') labeled with action $a \in A$ whenever either (i) $pre_a(v) = d$ and $eff_a(v) = d'$, or (ii) $pre_a(v)$ is not defined and $eff_a(v) = d'$.*
We refer to (d, d') as a value transition of v. We write $d \xrightarrow{a}_{\varphi} d'$ where $\varphi = pre_a \setminus \{v = d\}$ is the (outside) condition. Where not relevant, we omit "a" and/or "φ".

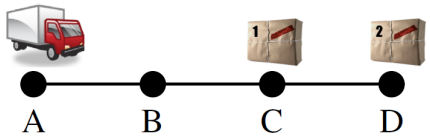
→ Captures "where a variable can go and how".

→ Attention: "value transition $d \xrightarrow{a}_{\varphi} d'$" $\neq$ "state transition $s \to s'$". (Value transition focuses on v, state transition encompasses all variables.)

Definition (Invertible Value Transition). *Let $\Pi = (V, A, c, I, G)$ be an FDR planning task, let $v \in V$, and let $d \to_{\varphi} d'$ be a value transition of v. We say that $d \to_{\varphi} d'$ is invertible if there exists a value transition $d' \to_{\varphi'} d$ where $\varphi' \subseteq \varphi$.*

→ Captures whether "we can go back".

Example: "Logistics"

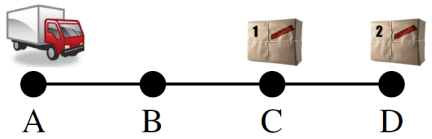


- State variables V: $truck : \{A, B, C, D\}$; $pack1, pack2 : \{A, B, C, D, T\}$.
- Actions A: $drive(x, y)$, $load(p, x)$, $unload(p, x)$. (Unit costs.)
- Initial state I: $truck = A$, $pack1 = C$, $pack2 = D$.
- goal G: $truck = A$, $pack1 = D$.

DTGs?

→ Are these value transitions invertible?
→ Example of non-invertible?

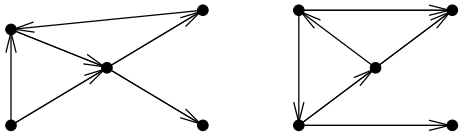
Why not in STRIPS?



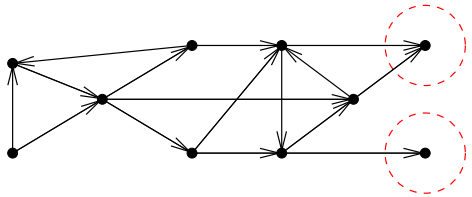
- Facts P: $truck(x)\, x \in \{A, B, C, D\}$; $pack1(x), pack2(x)\, x \in \{A, B, C, D, T\}$.
- Actions A: $drive(x, y)$, $load(p, x)$, $unload(p, x)$. (Unit costs.)
- Initial state I: $\{truck(A), pack1(C), pack2(D)\}$.
- goal G: $\{truck(A), pack1(D)\}$.

DTGs?

Task Decomposition: Unconnected Sub-Tasks



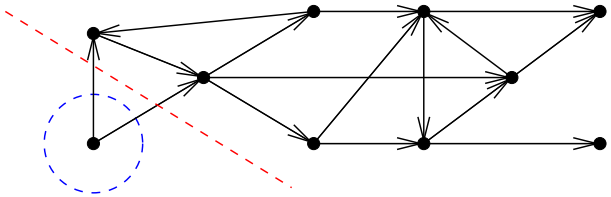
Task Simplification: Non-Goal Leaf Variables



Question!

How can we simplify Π if there is a leaf vertex v on which $G(v)$ is undefined?

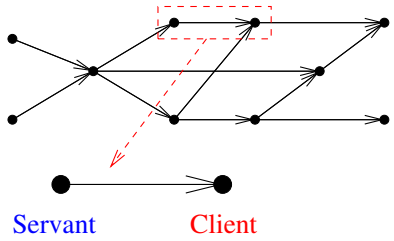
Task Simplification: Invertible Root Variables



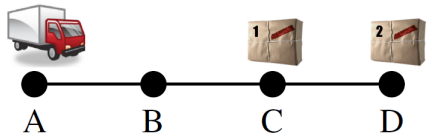
Task Simplification: Invertible Root Variables, ctd.

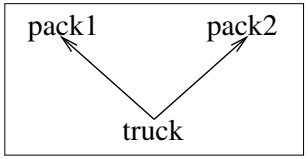
Complexity: Acyclic + Invertible

Servants + clients, now in full:



The Plan Construction in "Logistics"





- Initial state I: $truck = A$, $pack1 = C$, $pack2 = D$.
- goal G: $truck = A$, $pack1 = D$, $pack2 = A$.

→ Topological order: $v_1 = truck$, $v_2 = pack2$, $v_3 = pack1$.
→ Targets for $pack1$:
→ Path for $pack1$:
→ Targets for $pack2$:
→ Path for $pack2$:
→ Targets for $truck$:
→ Path for $truck$:

Complexity: Acyclic + Invertible, ctd.

Theorem. *Restrict the input to FDR tasks $\Pi = (V, A, c, I, G)$ such that $CG(\Pi)$ is acyclic and, for all $v \in V$, all value transitions of v are invertible. Then PlanEx can be decided in polynomial time.*

→ Note: We do *not* require the DTGs to be connected here. If they were connected, Π would be solvable and there would be no PlanEx to decide. Also, Π can be solvable even for non-connected DTGs:

Summary

- For general problem solving to be effective, it is essential to automatically detect and exploit problem structure.
- Causal graphs are the most prominent means to capture problem structure in planning; they are typically considered along with domain transition graphs.
- Causal graphs can be used for a variety of purposes, including task decomposition/simplification and complexity analysis.
- Simple decomposition/simplification methods are to split up unconnected components, remove invertible root variables, remove non-goal leaf variables.

  → We will rely on some of this in **Chapter 12**.
- One tractable class is the special case where the causal graph is acyclic and all value transitions are invertible.

  → We will consider that special case again in partial delete relaxation **(Chapter 10)**.

Reading

- *The Fast Downward Planning System* [Helmert (2006)].

  Available at:

  `http://www.jair.org/media/1705/live-1705-2731-jair.pdf`

  Content: This is the initial paper on the Fast Downward planning system, which in the meantime has grown into the main implementation basis for heuristic search planning. I suggest it for this chapter because it very clearly compares causal graphs for STRIPS vs. those for FDR (FDR is called "SAS+" and "multi-valued planning" in there), and to my knowledge it's the first paper introducing DTGs. The part of the paper up to Section 5.2 (first 25 pages) is directly relevant to the present chapter; the remainder of the paper is not, but is definitely useful background knowledge for heuristic search planning, and thus for this course as a whole.

References I

Ronen Brafman and Carmel Domshlak. Structure and complexity in planning with unary operators. *Journal of Artificial Intelligence Research*, 18:315–349, 2003.

Ronen Brafman and Carmel Domshlak. On the complexity of planning for agent teams and its implications for single agent planning. *Artificial Intelligence*, 198:52–71, 2013.

Hubie Chen and Omer Giménez. Causal graphs and structurally restricted planning. *Journal of Computer and System Sciences*, 76(7):579–592, 2010.

Carmel Domshlak and Yefim Dinitz. Multi-agent offline coordination: Structure and complexity. In A. Cesta and D. Borrajo, editors, *Proceedings of the 6th European Conference on Planning (ECP'01)*, pages 34–43. Springer-Verlag, 2001.

Carmel Domshlak, Jörg Hoffmann, and Michael Katz. Red-black planning: A new systematic approach to partial delete relaxation. *Artificial Intelligence*, 221:73–114, 2015.

Omer Giménez and Anders Jonsson. Planning over chain causal graphs for variables with domains of size 5 is NP-hard. *Journal of Artificial Intelligence Research*, 34:675–706, 2009.

References II

Daniel Gnad and Jörg Hoffmann. Beating LM-cut with h^{max} (sometimes): Fork-decoupled state space search. In Ronen Brafman, Carmel Domshlak, Patrik Haslum, and Shlomo Zilberstein, editors, *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS'15)*, pages 88–96. AAAI Press, 2015.

Malte Helmert. A planning heuristic based on causal graph analysis. In Sven Koenig, Shlomo Zilberstein, and Jana Koehler, editors, *Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS'04)*, pages 161–170, Whistler, Canada, 2004. Morgan Kaufmann.

Malte Helmert. The Fast Downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

Jörg Hoffmann. Analyzing search topology without running any search: On the connection between causal graphs and h^+. *Journal of Artificial Intelligence Research*, 41:155–229, 2011.

Michael Katz and Carmel Domshlak. New islands of tractability of cost-optimal planning. *Journal of Artificial Intelligence Research*, 32:203–288, 2008.

References III

Michael Katz and Carmel Domshlak. Implicit abstraction heuristics. *Journal of Artificial Intelligence Research*, 39:51–126, 2010.

Michael Katz and Emil Keyder. Structural patterns beyond forks: Extending the complexity boundaries of classical planning. In Jörg Hoffmann and Bart Selman, editors, *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI'12)*, pages 1779–1785, Toronto, ON, Canada, July 2012. AAAI Press.

Craig Knoblock. Automatically generating abstractions for planning. *Artificial Intelligence*, 68(2):243–302, 1994.

Brian C. Williams and P. P. Nayak. A reactive planner for a model-based executive. In M. Pollack, editor, *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 1178–1185, Nagoya, Japan, August 1997. Morgan Kaufmann.