Finding User-friendly Linearizations of Partially Ordered Plans

Daniel Höller and Pascal Bercher and Felix Richter and Marvin Schiller and Thomas Geier and Susanne Biundo

Institute of Artificial Intelligence

September 23rd, 2014

ulm university universität ulm

Provide advanced user assistance based on:

- *user-centered* planning:
  plan generation, execution, repair, explanation
- user interaction:
  dialog and interaction management

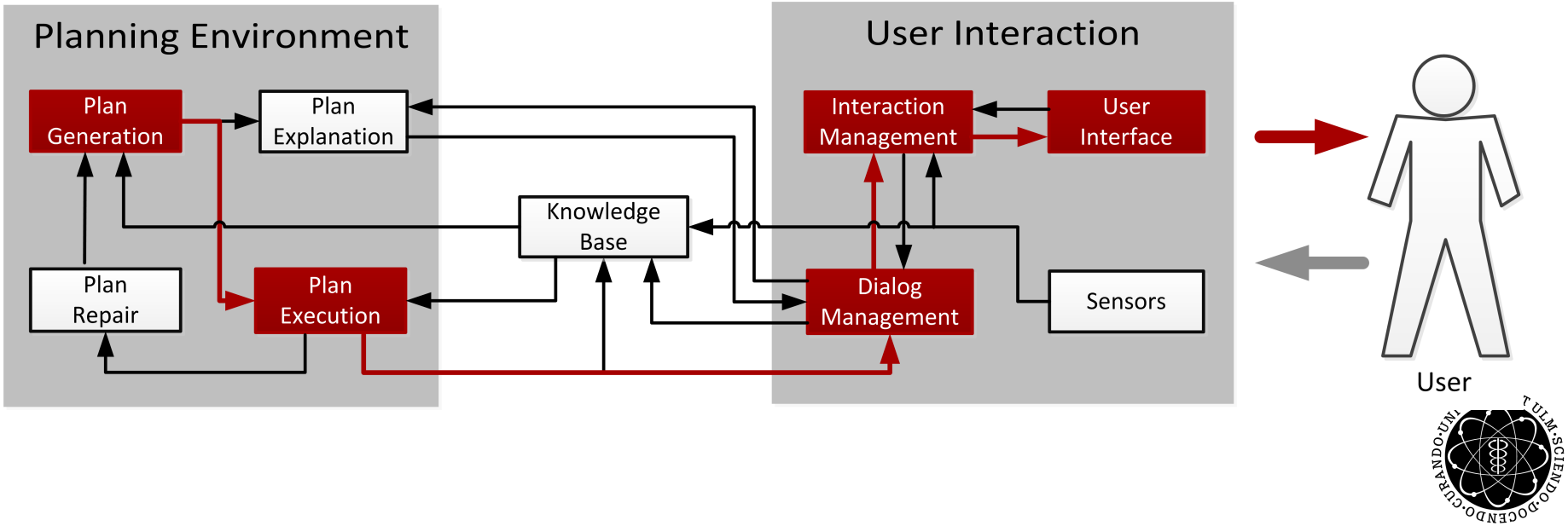
Example domain:

- set up a complex home theater

The Assembly Task:

communicate solution plan to the user:

- present the solution plan action by action
- display each primitive action in an adequate manner
  - load dialog model for each action
  - display dialog according to interaction management

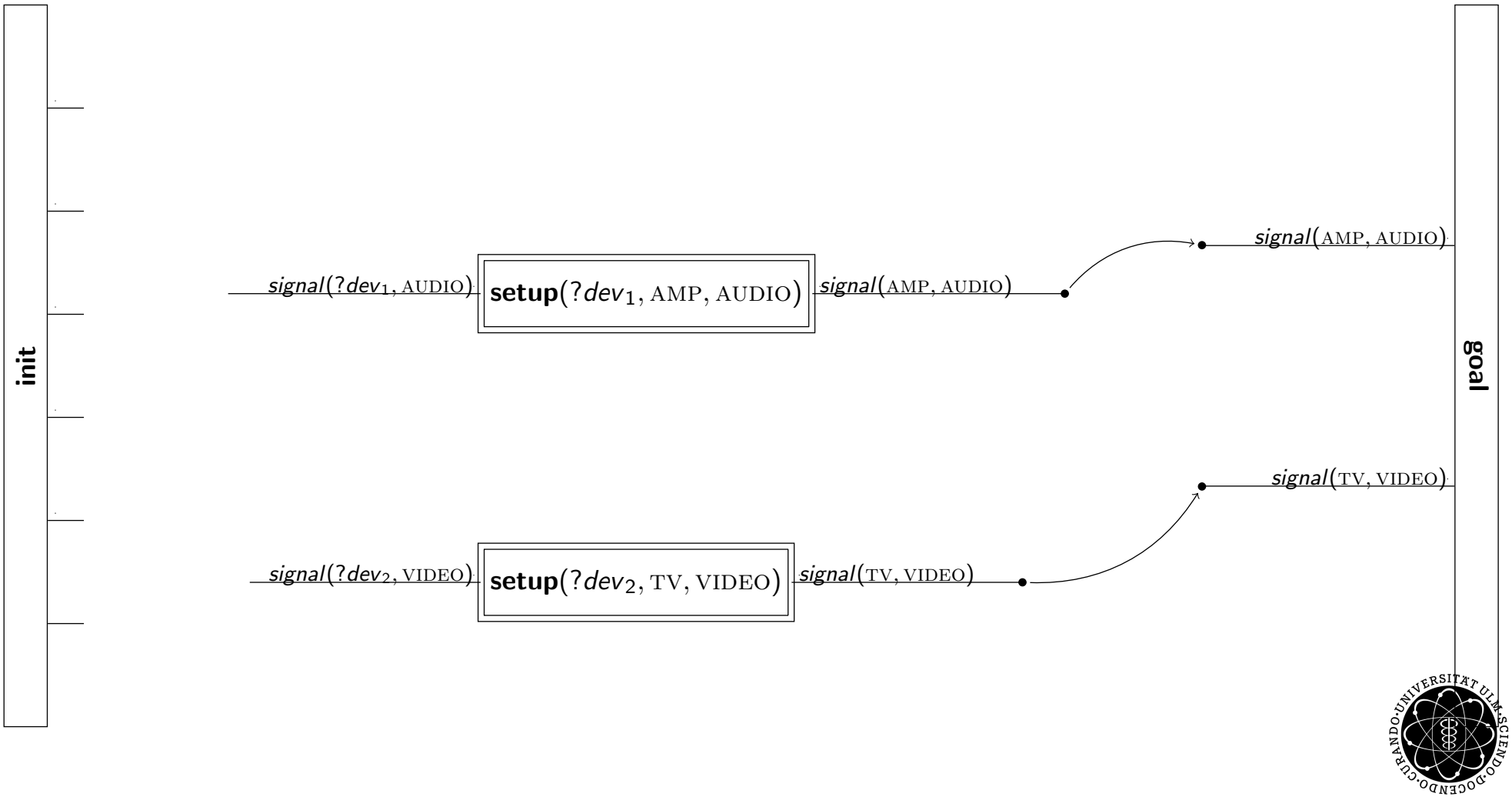


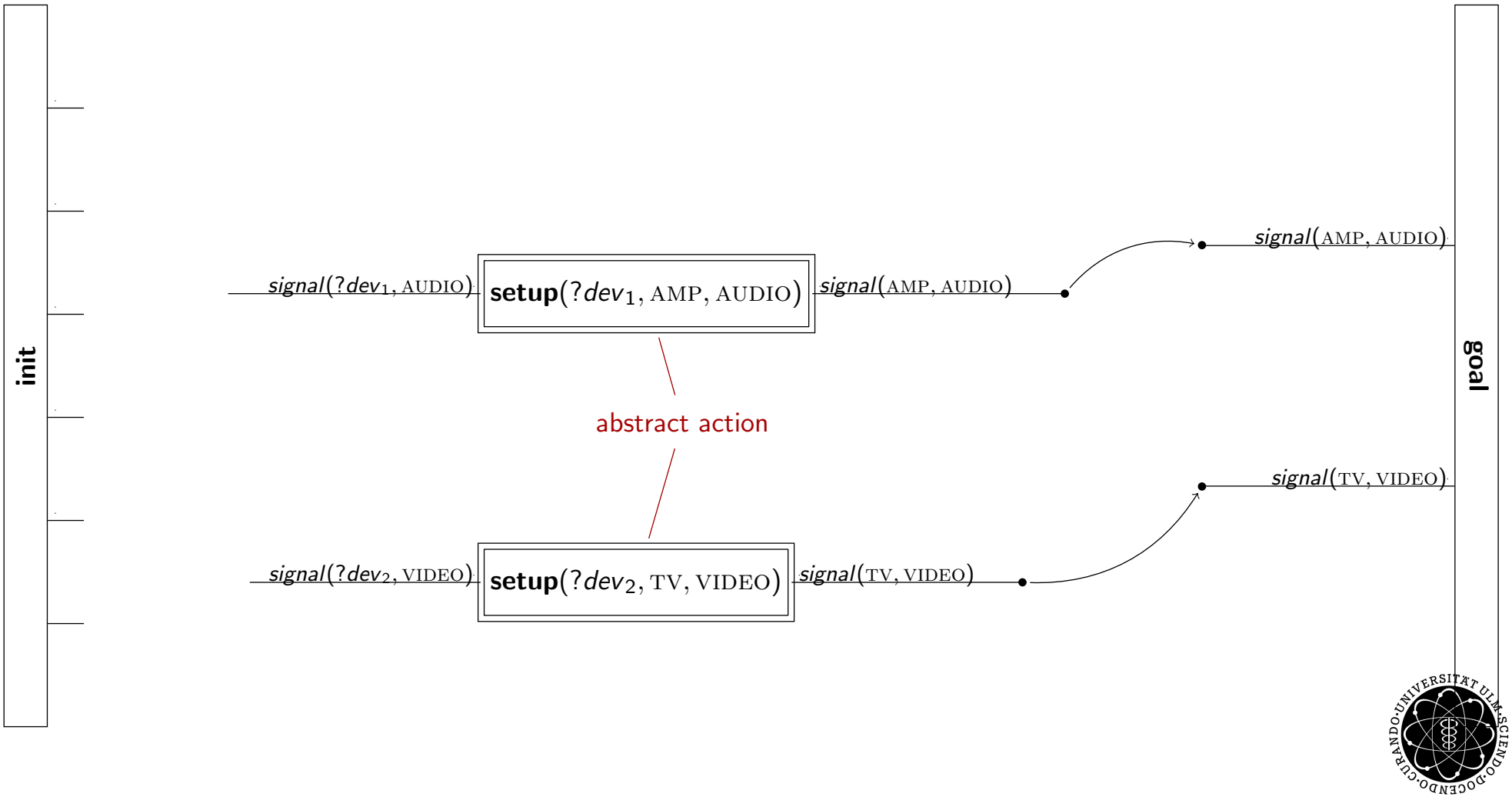
Why bother about plan linearization?

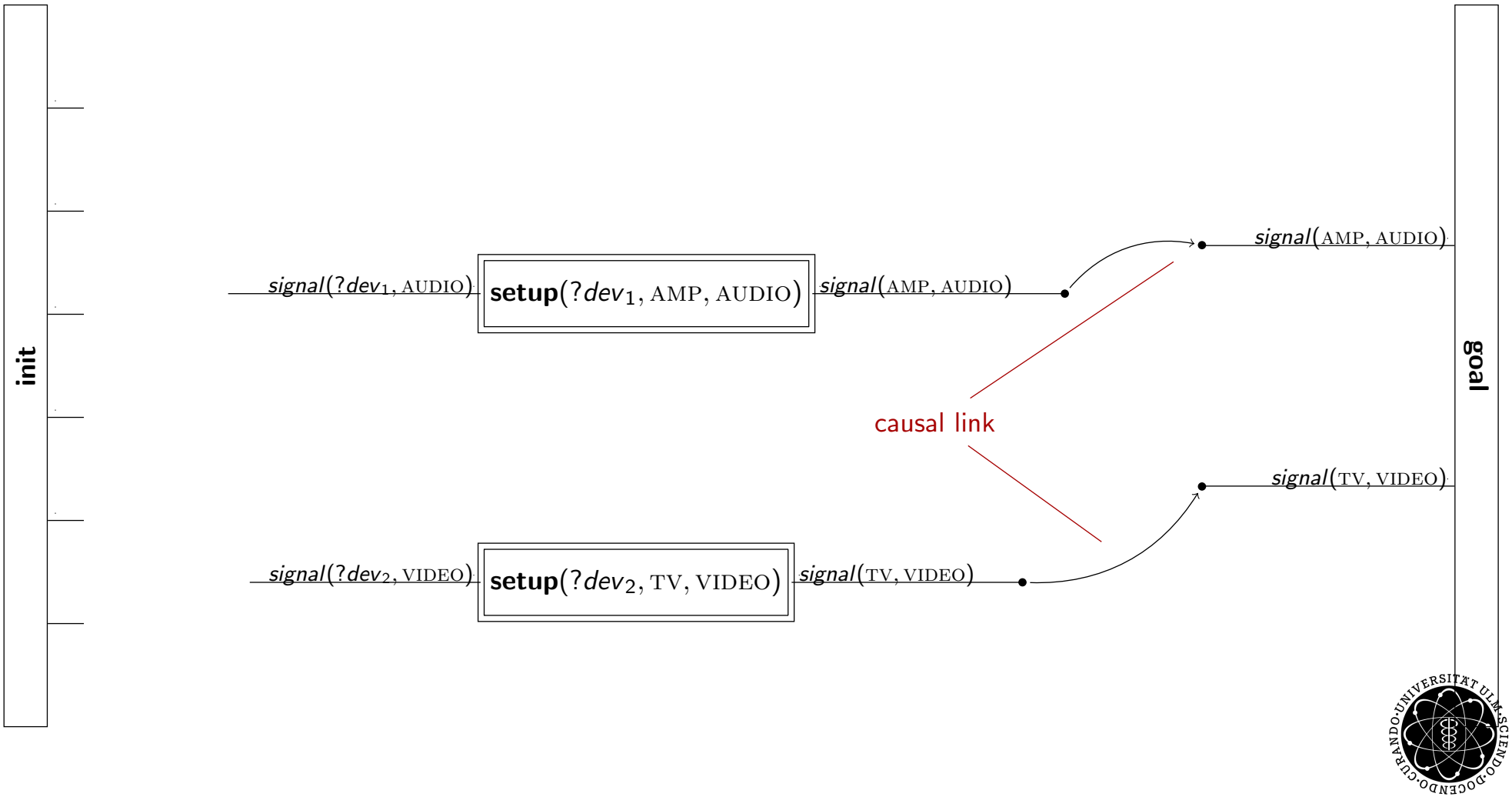
- planning systems return plans that induce executable sequences of primitive tasks
- plans are executed by a human user
- planning problem and solution include information that can be exploited
- we give three domain-independent approaches and illustrate them in the “home theater” example domain

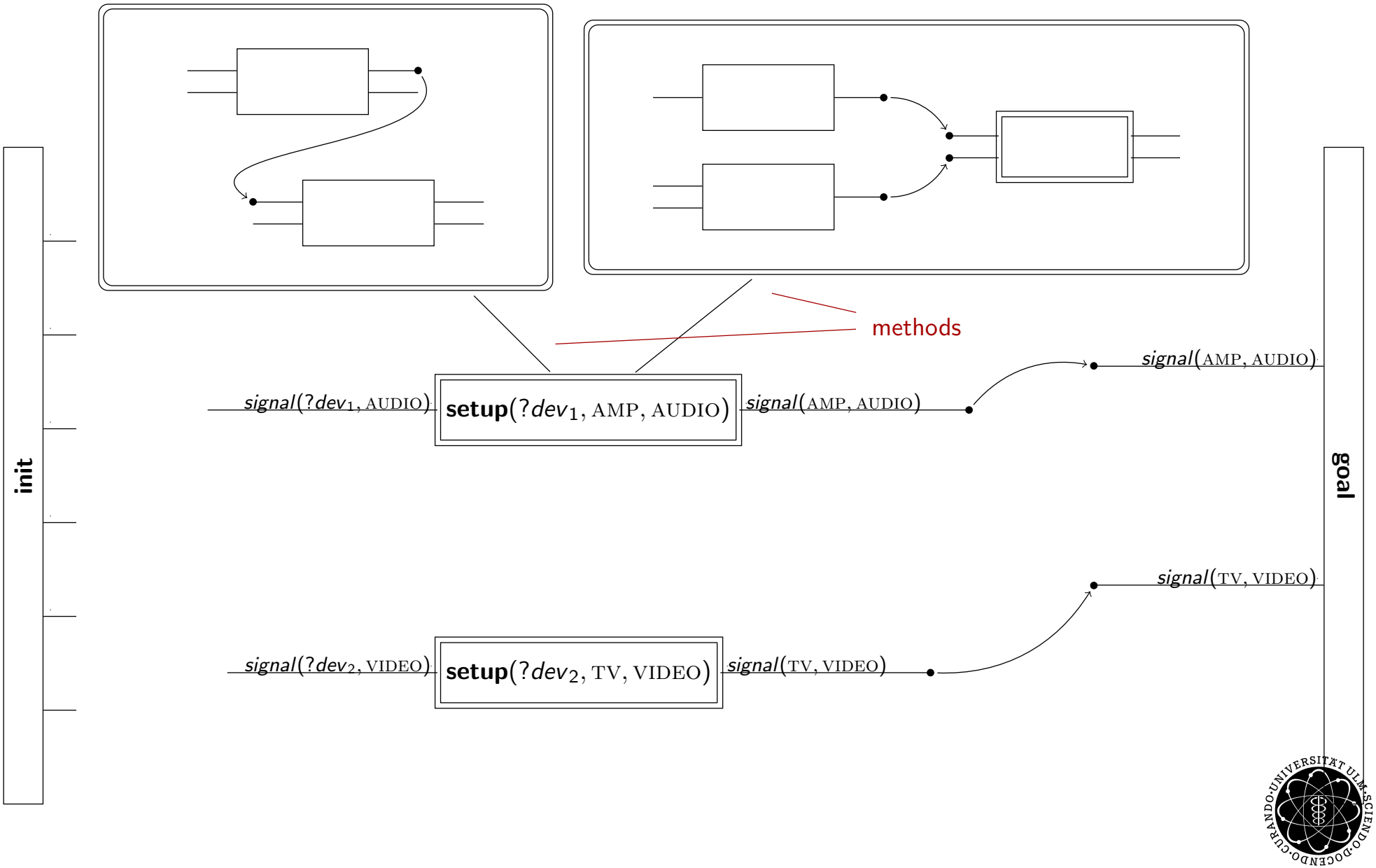
Hybrid Planning:

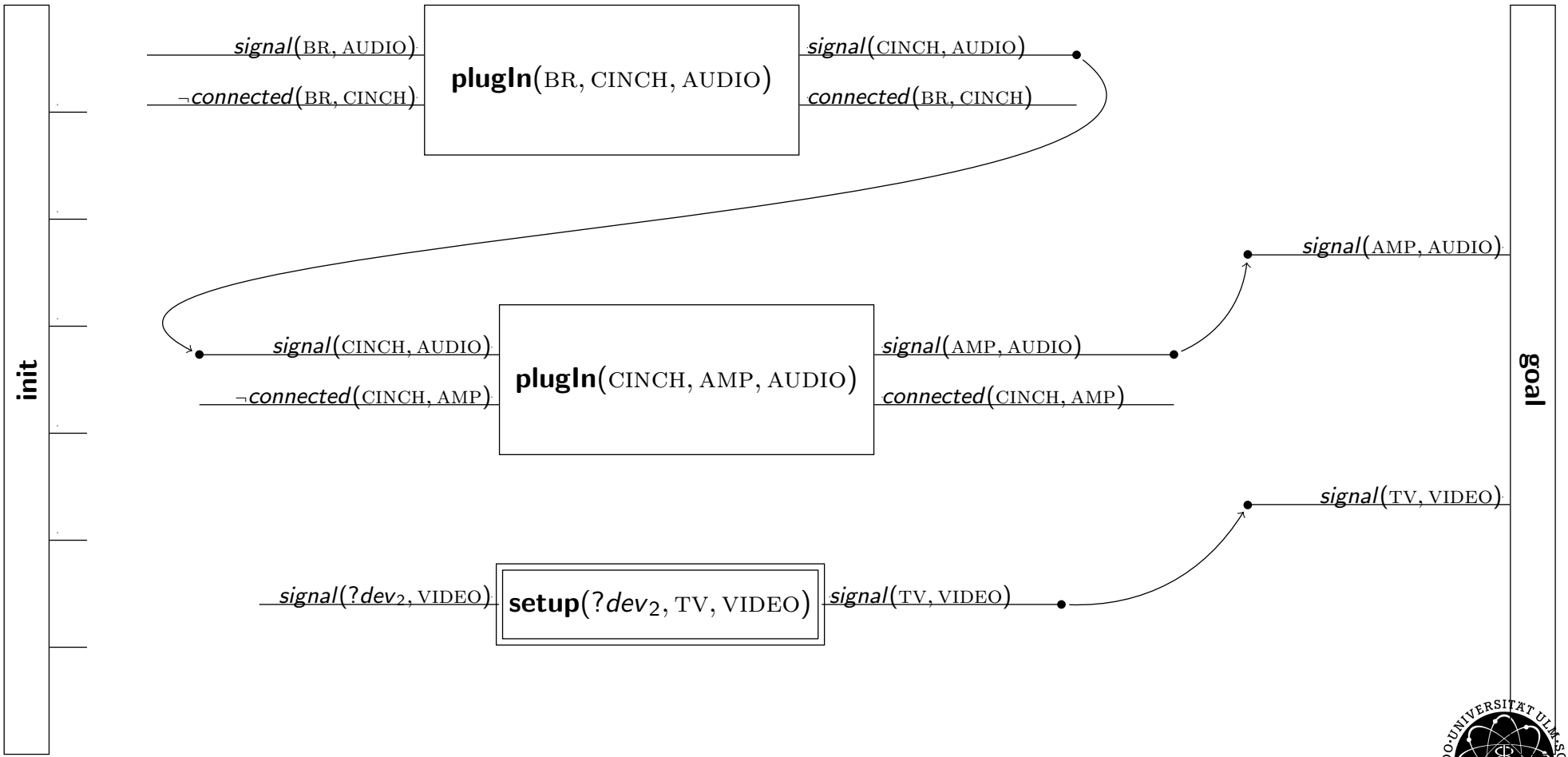
- approach fusing Hierarchical Task Network (HTN) Planning with Partial-Order Causal-Link (POCL) Planning
- search in the space of partial plans
- refine the initial partial plan until it is executable

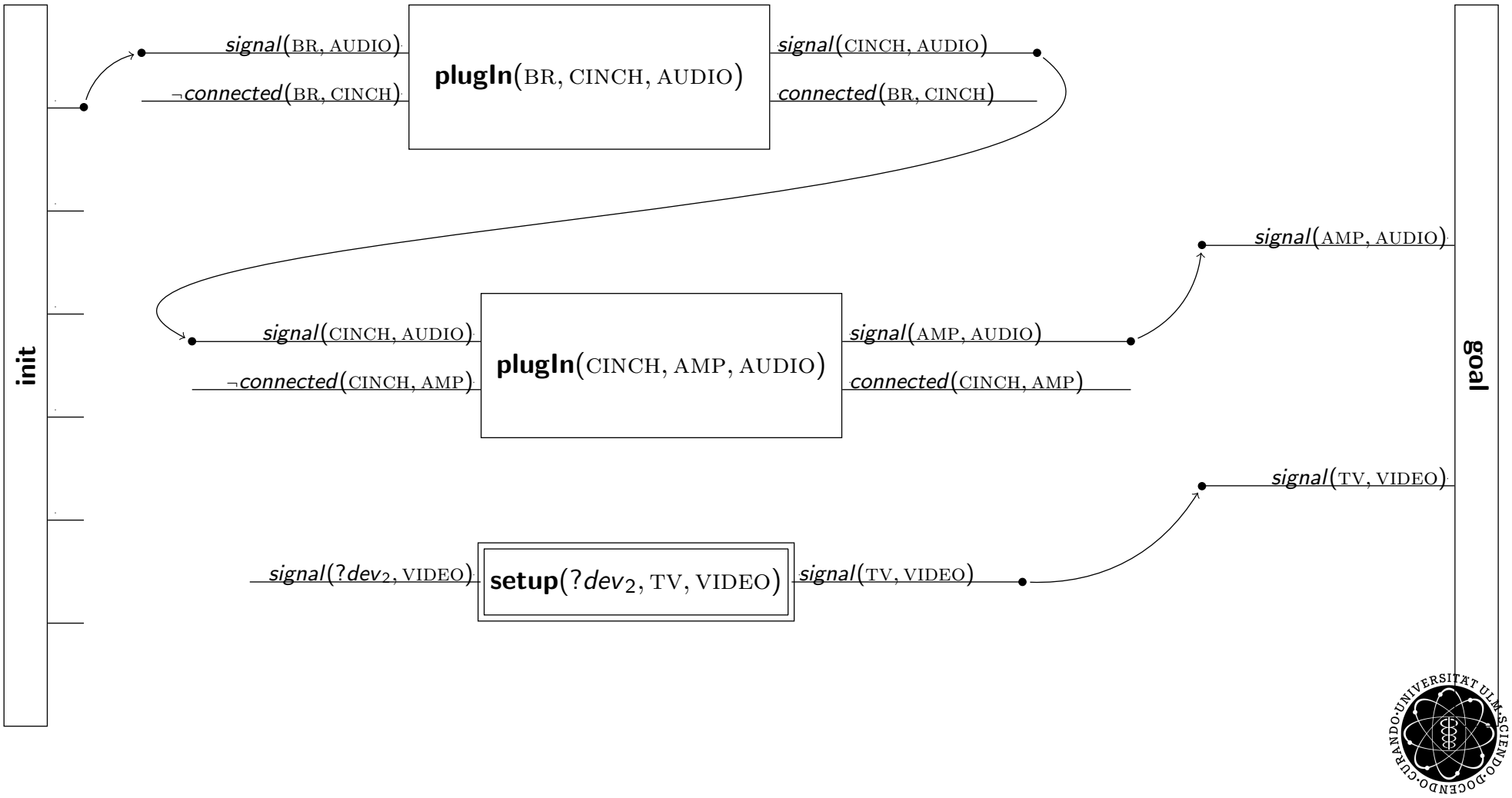

init
goal
signal(?dev1, AUDIO)
setup(?dev1, AMP, AUDIO)
signal(AMP, AUDIO)
signal(AMP, AUDIO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)

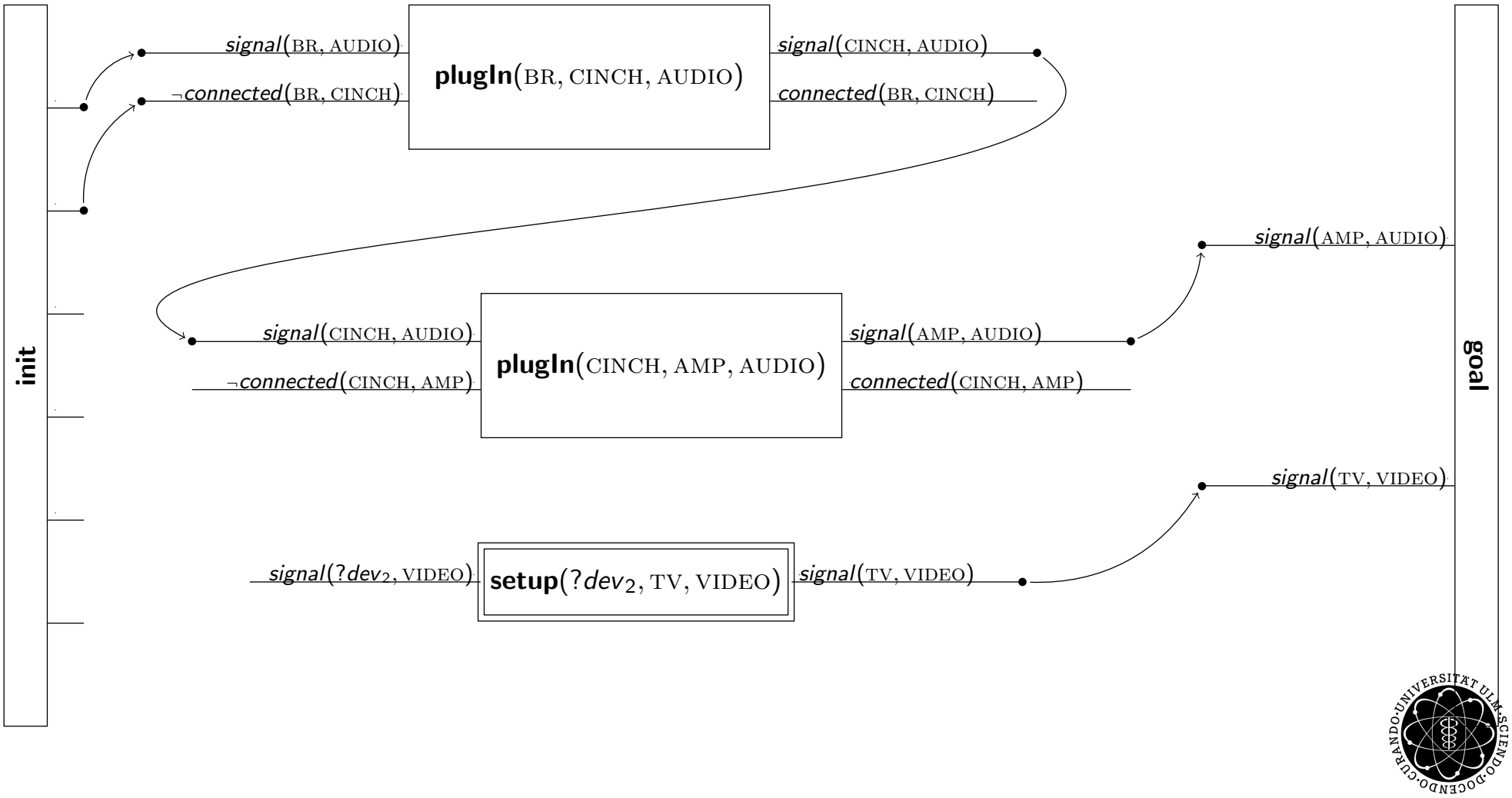

init
goal
signal(?dev1, AUDIO)
setup(?dev1, AMP, AUDIO)
signal(AMP, AUDIO)
signal(AMP, AUDIO)
abstract action
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)

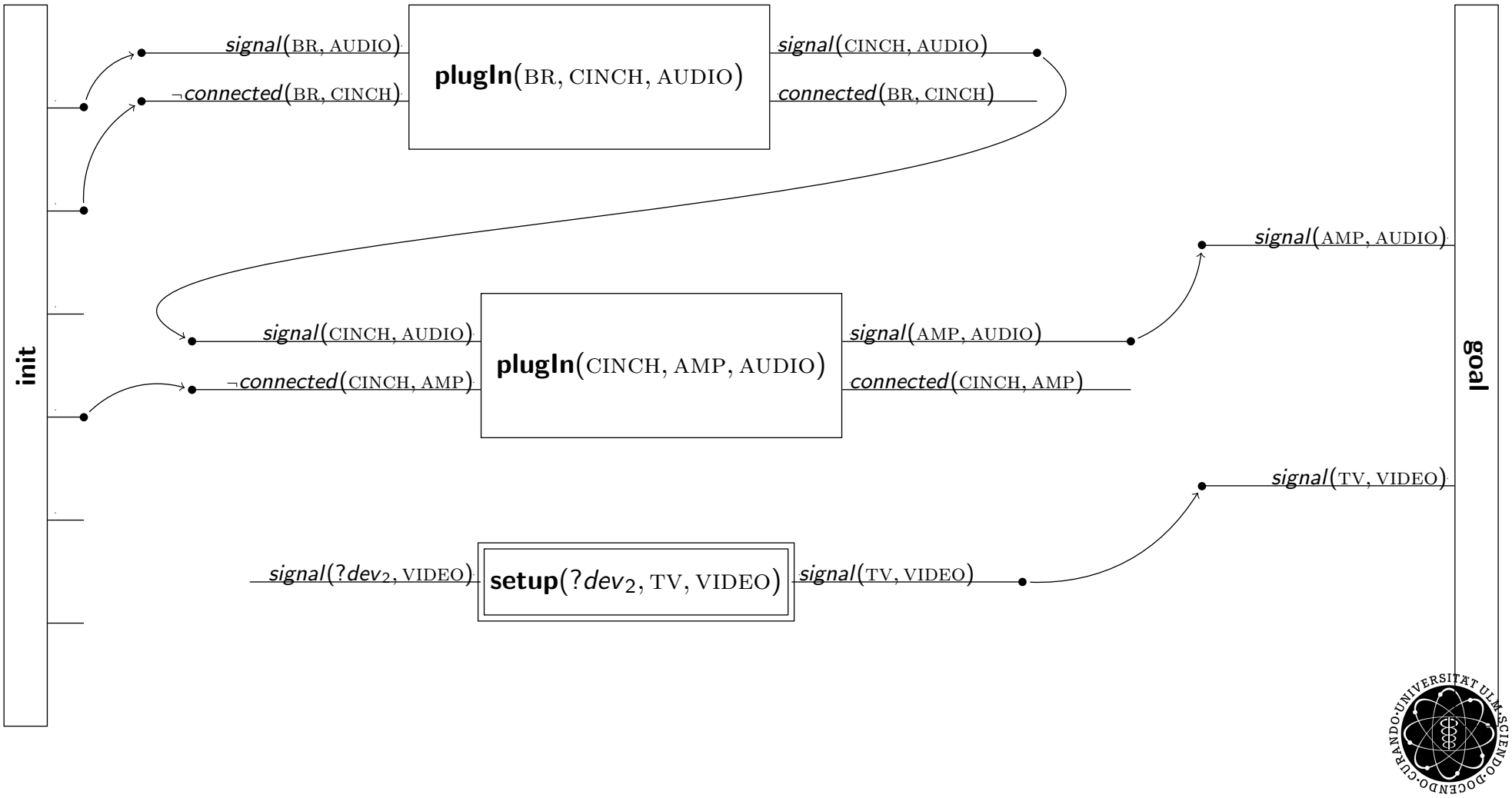

init
goal
signal(?dev1, AUDIO)
setup(?dev1, AMP, AUDIO)
signal(AMP, AUDIO)
signal(AMP, AUDIO)
causal link
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)

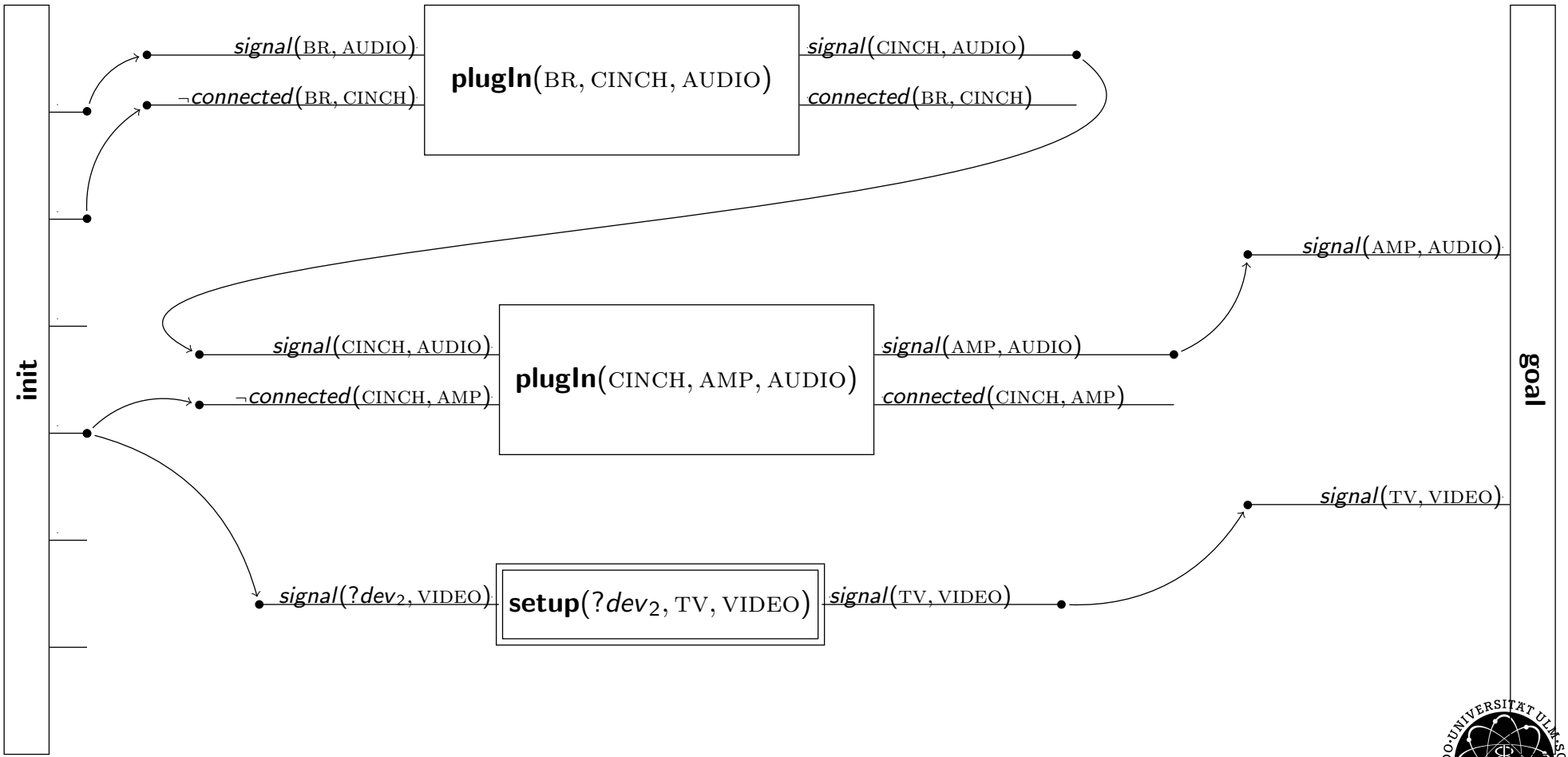

methods
init
goal
signal(?dev1, AUDIO)
setup(?dev1, AMP, AUDIO)
signal(AMP, AUDIO)
signal(AMP, AUDIO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)

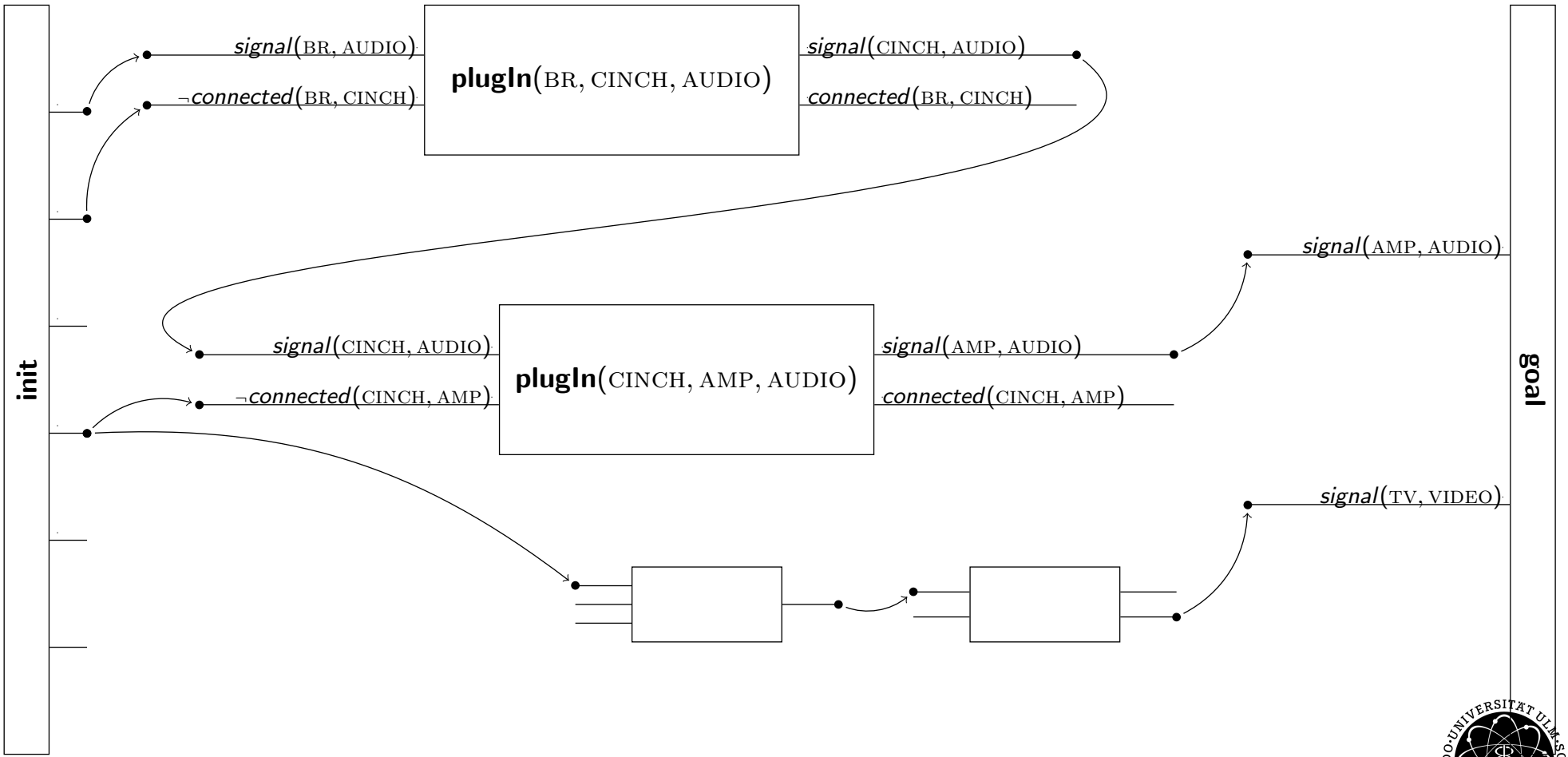

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(TV, VIDEO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
goal

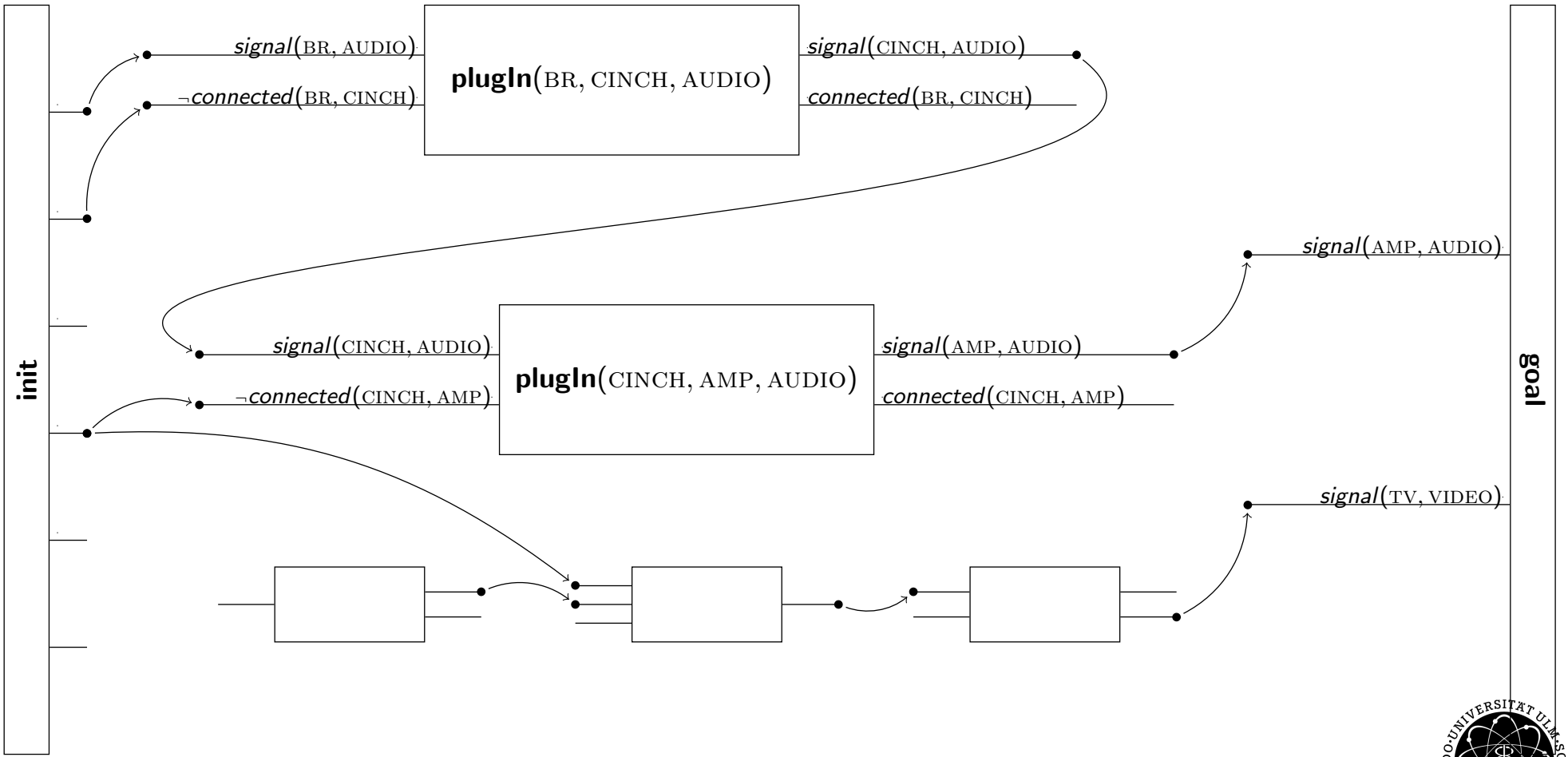

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(TV, VIDEO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
goal

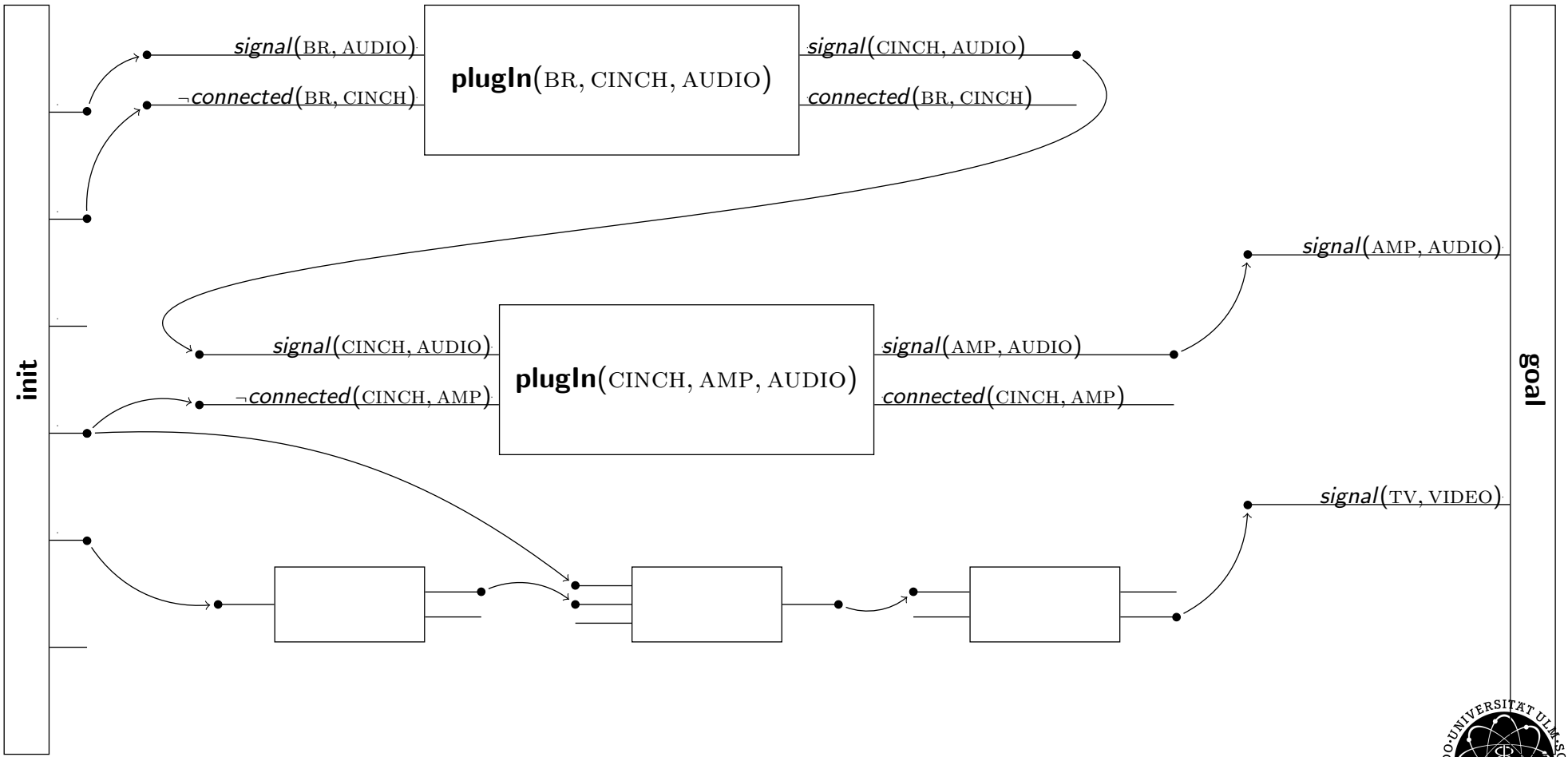

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(AMP, AUDIO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)
goal

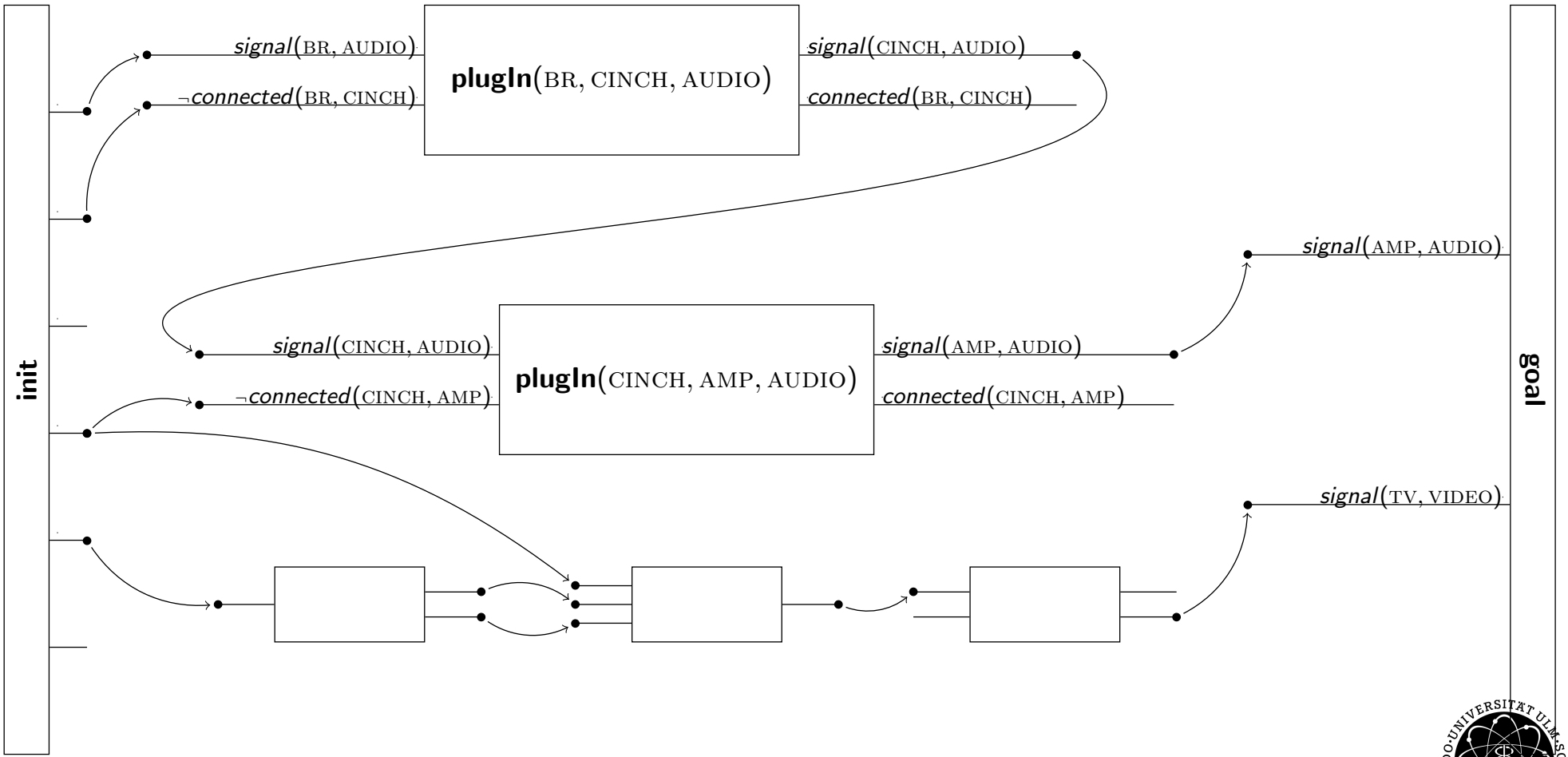

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(TV, VIDEO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
goal

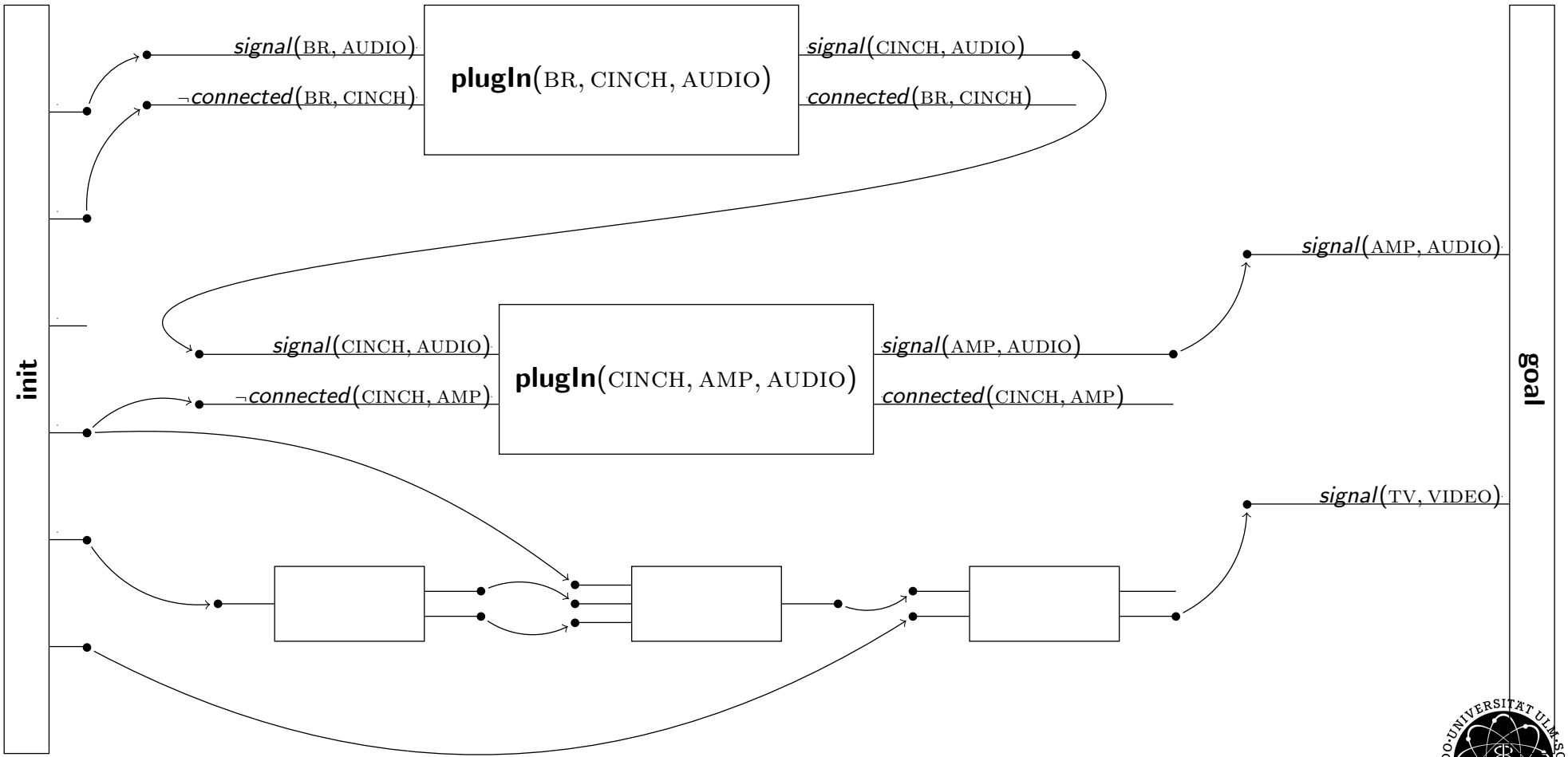

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
goal
signal(TV, VIDEO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)


init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(TV, VIDEO)
goal

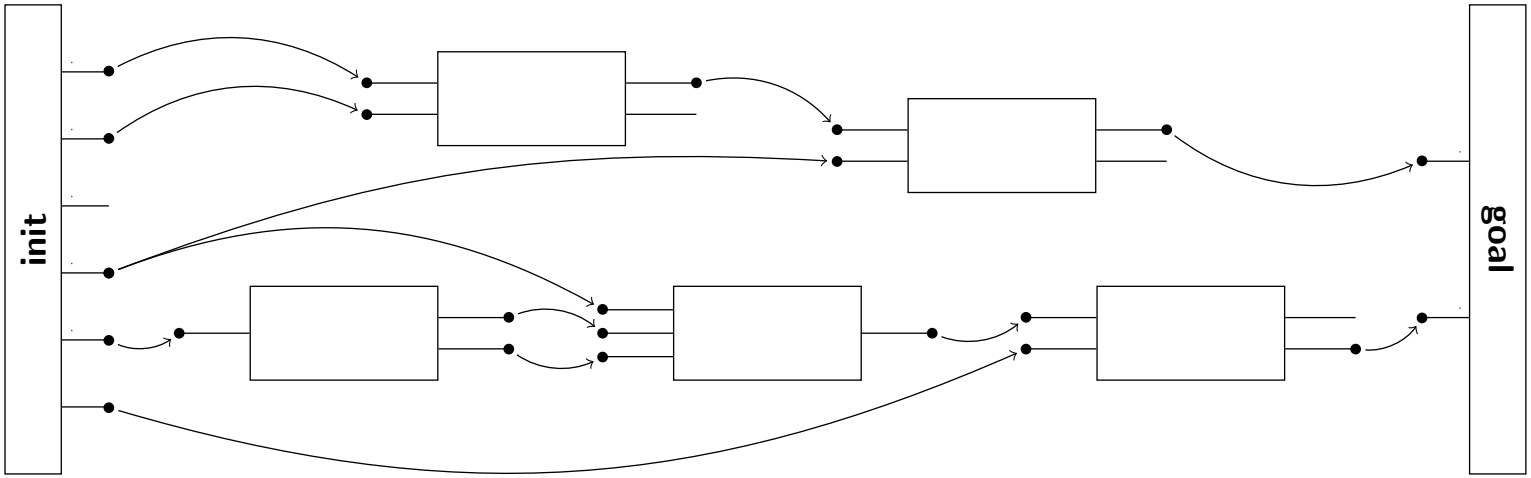

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(TV, VIDEO)
goal


init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(TV, VIDEO)
goal

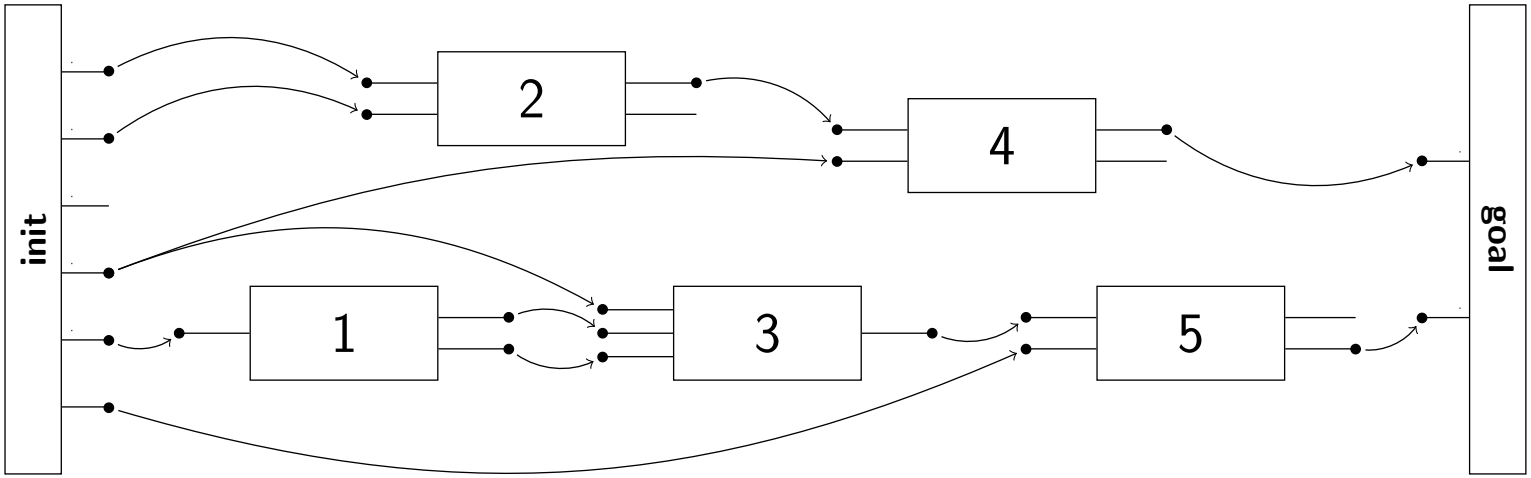

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(TV, VIDEO)
goal


init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AMP, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AMP)
plugIn(CINCH, AMP, AUDIO)
signal(AMP, AUDIO)
connected(CINCH, AMP)
signal(TV, VIDEO)
goal

generate a plausible linearization of the actions:

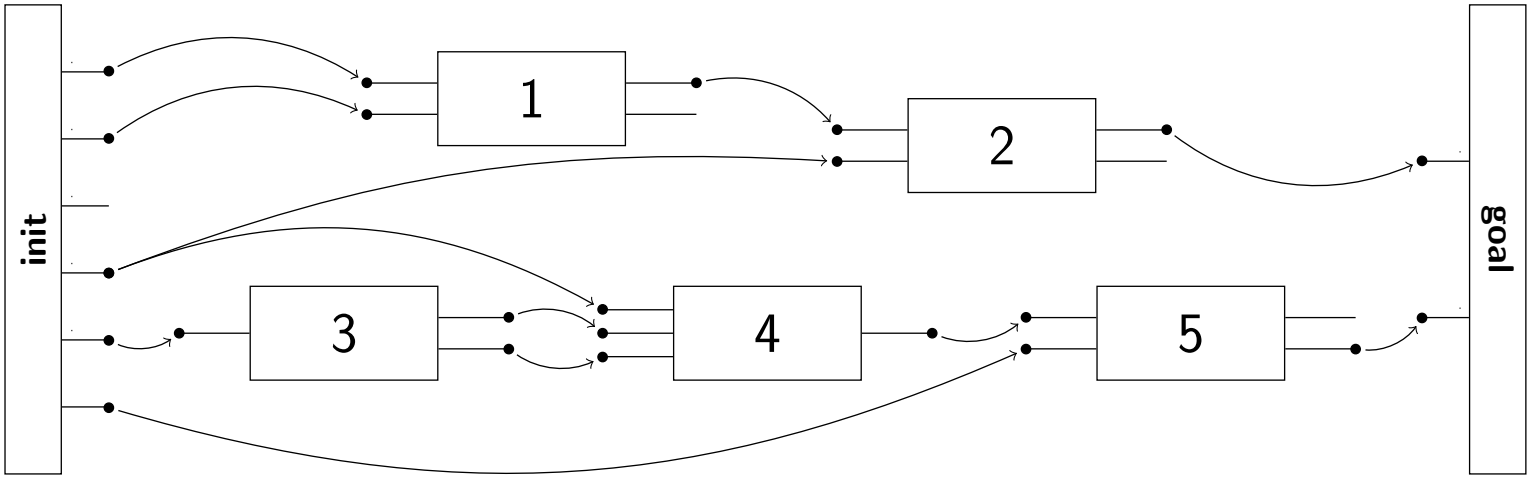


generate a plausible linearization of the actions:



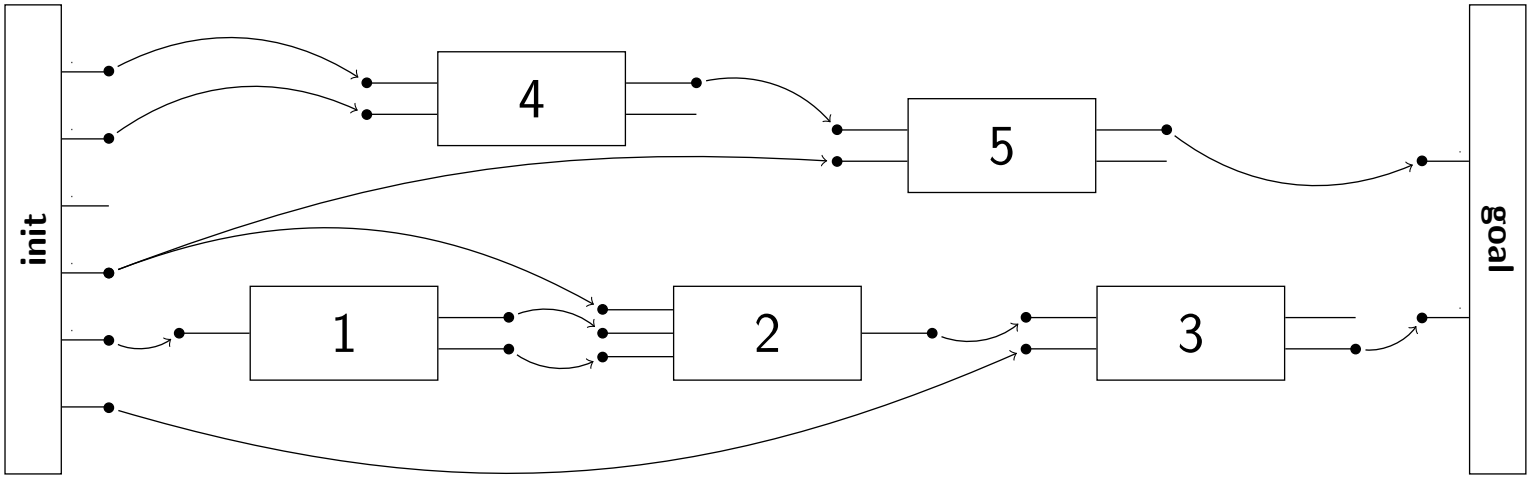
1: connect ...
2: connect CINCH cable (the first end) with Blu-ray player
3: connect ...
4: connect CINCH cable (the other end) with AV receiver
5: connect ...

generate a plausible linearization of the actions:



1: connect CINCH cable (the first end) with Blu-ray player
2: connect CINCH cable (the other end) with AV receiver
3: connect ...
4: connect ...
5: connect ...

generate a plausible linearization of the actions:



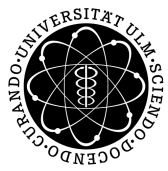
1: connect ...
2: connect ...
3: connect ...
4: connect CINCH cable (the first end) with Blu-ray player
5: connect CINCH cable (the other end) with AV receiver

information used for finding user-friendly plan linearizations:

- the planning domain and
- the solution to the given planning problem

we introduce three domain-independent linearizations approaches based on:

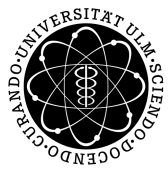
- **action parameters**
- **causal links** in the plan
- **decomposition hierarchy**



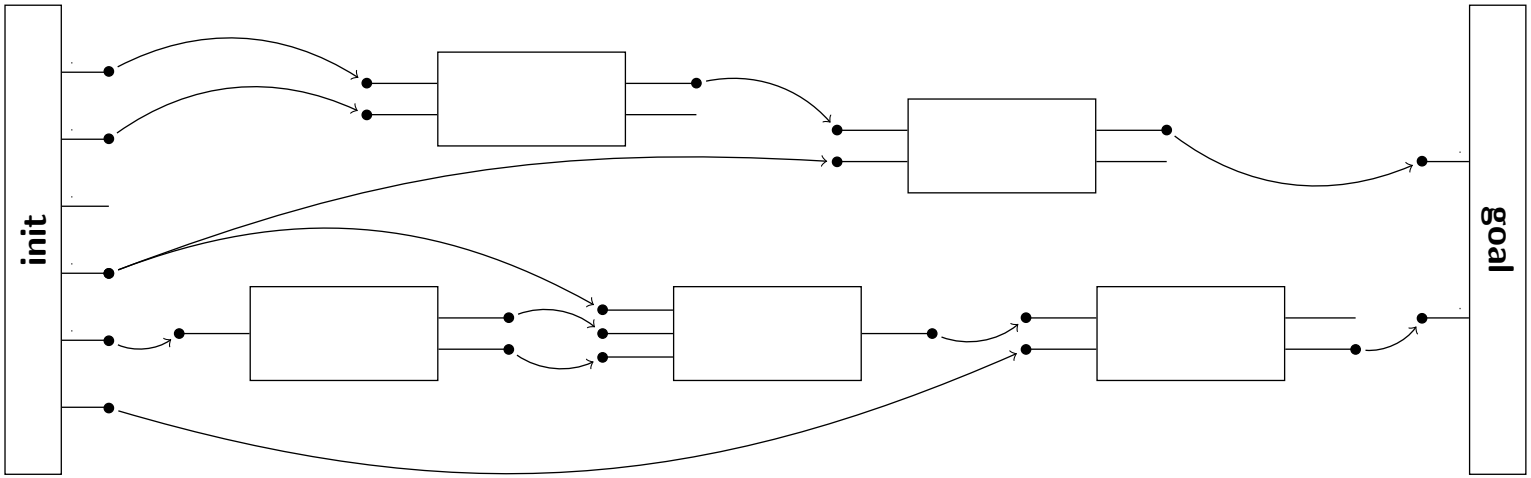
Parameter-based plan linearization

- actions represent activities to do
- parameters introduce the items/objects/subjects to use

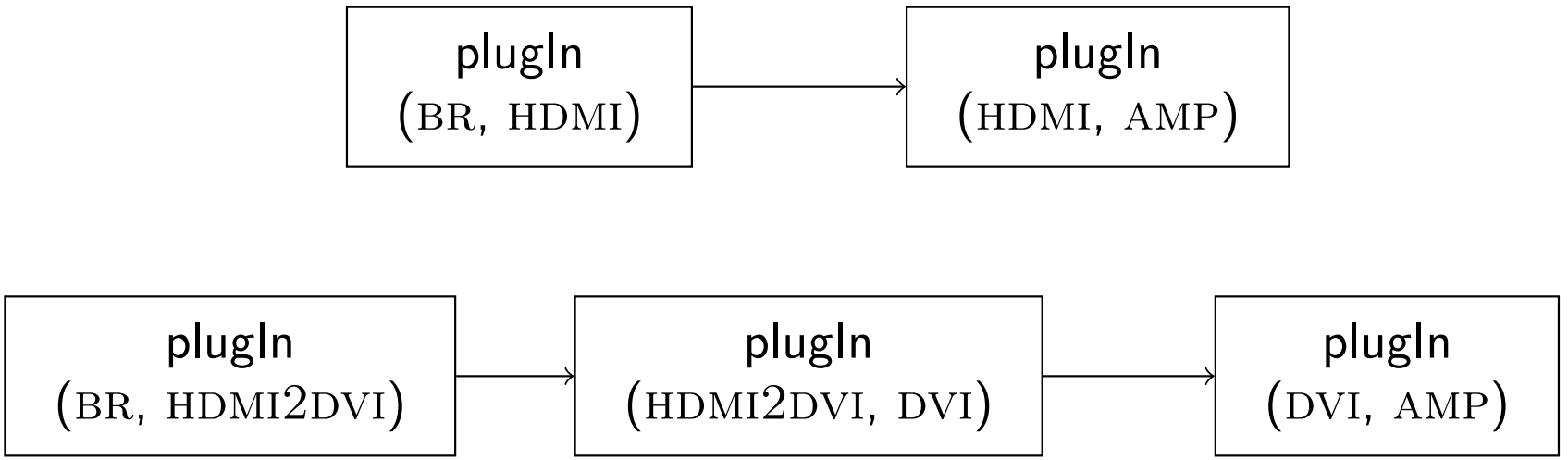
→ execute actions involving the same parameters consecutively



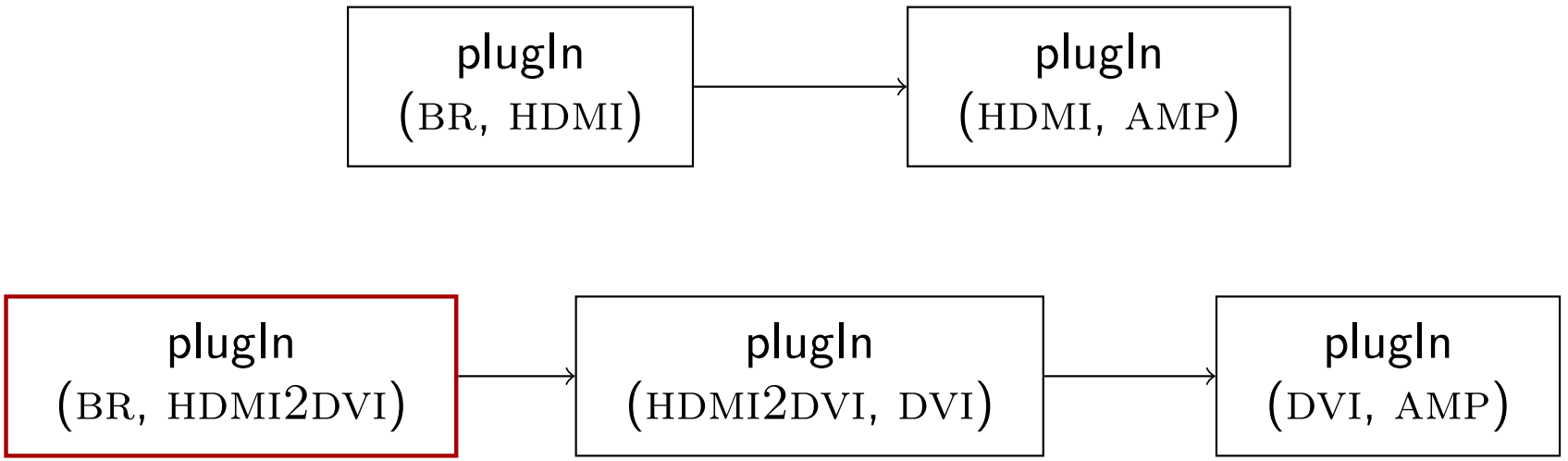
Solution plan (schematically, with causal structure)



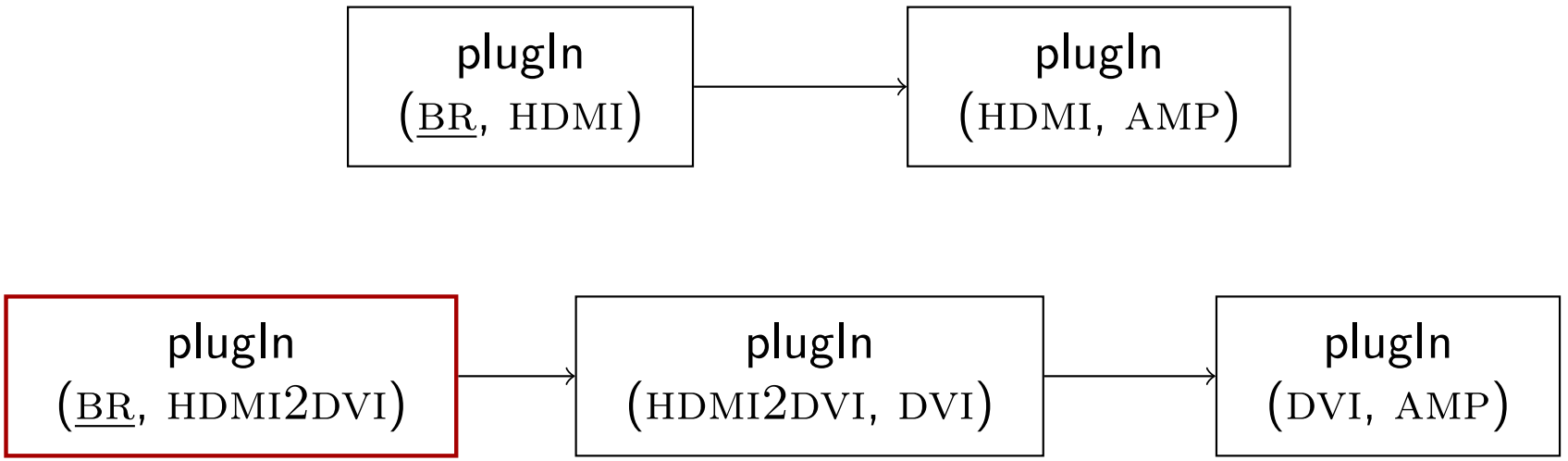
Solution plan (ordering constraints, action schemata)



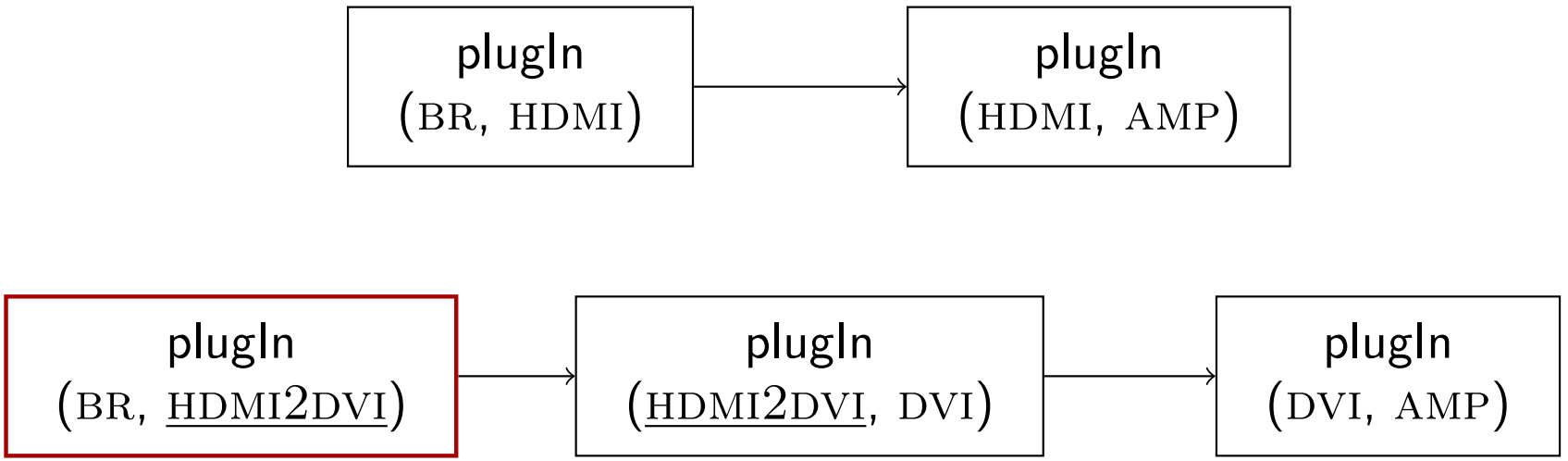
Solution plan (ordering constraints, action schemata)



Solution plan (ordering constraints, action schemata)



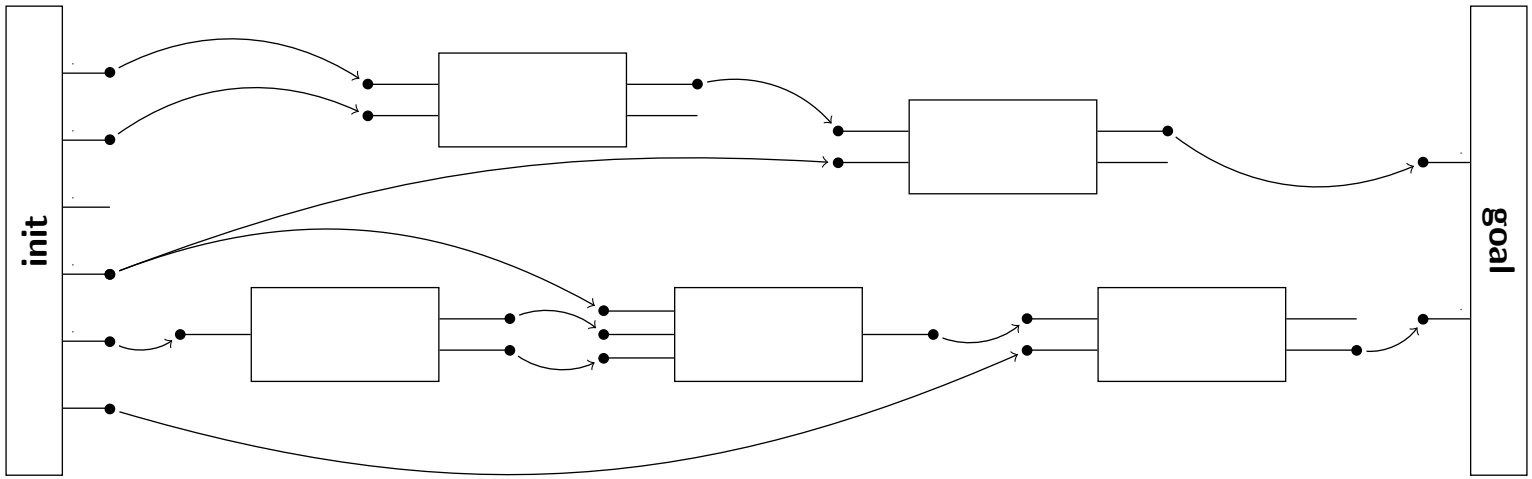
Solution plan (ordering constraints, action schemata)



Causal link-based plan linearization

- causal links explicitly represent the causal structure of the plan
- each link was introduced to solve a flaw – all links are required

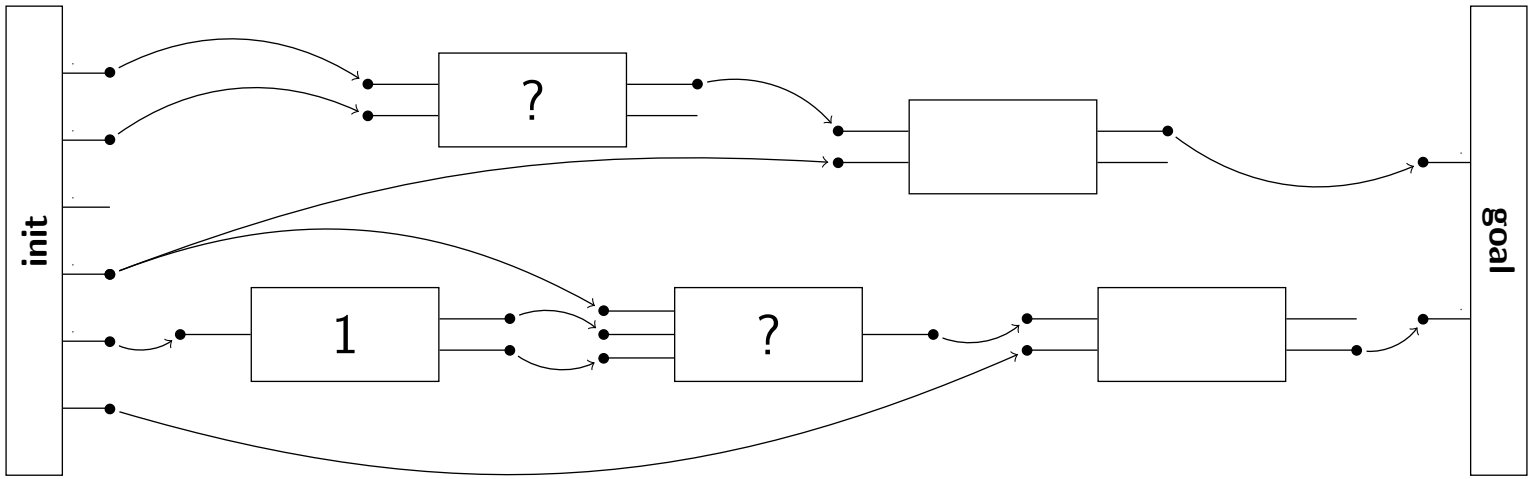
→ execute connected actions consecutively



Causal link-based plan linearization

- causal links explicitly represent the causal structure of the plan
- each link was introduced to solve a flaw – all links are required

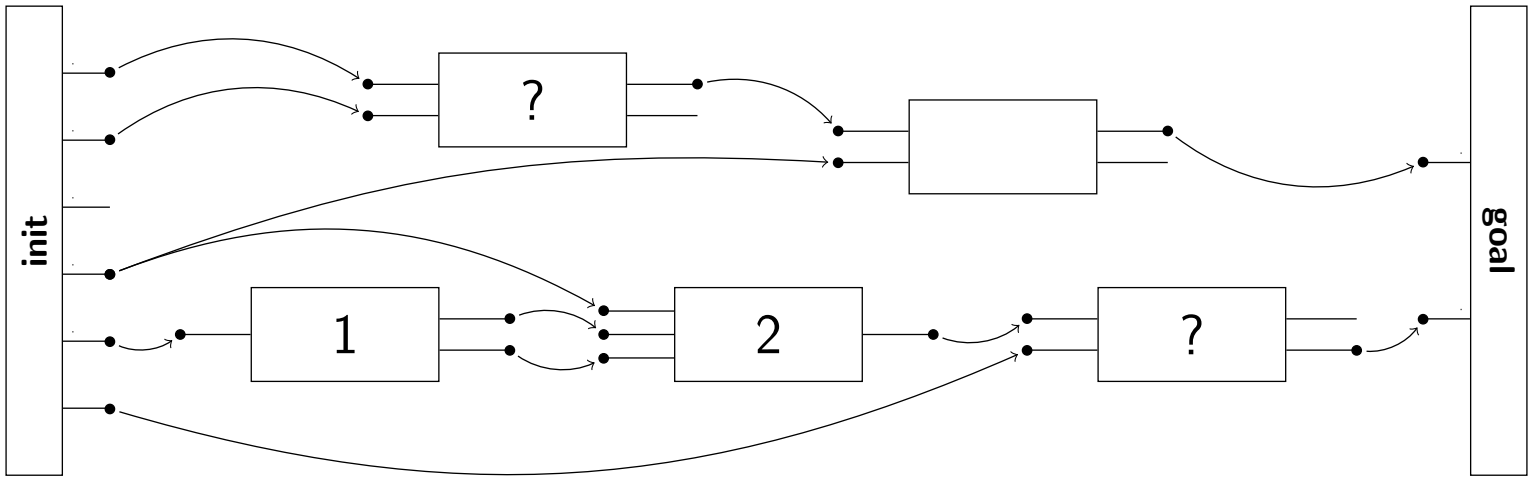
→ execute connected actions consecutively



Causal link-based plan linearization

- causal links explicitly represent the causal structure of the plan
- each link was introduced to solve a flaw – all links are required

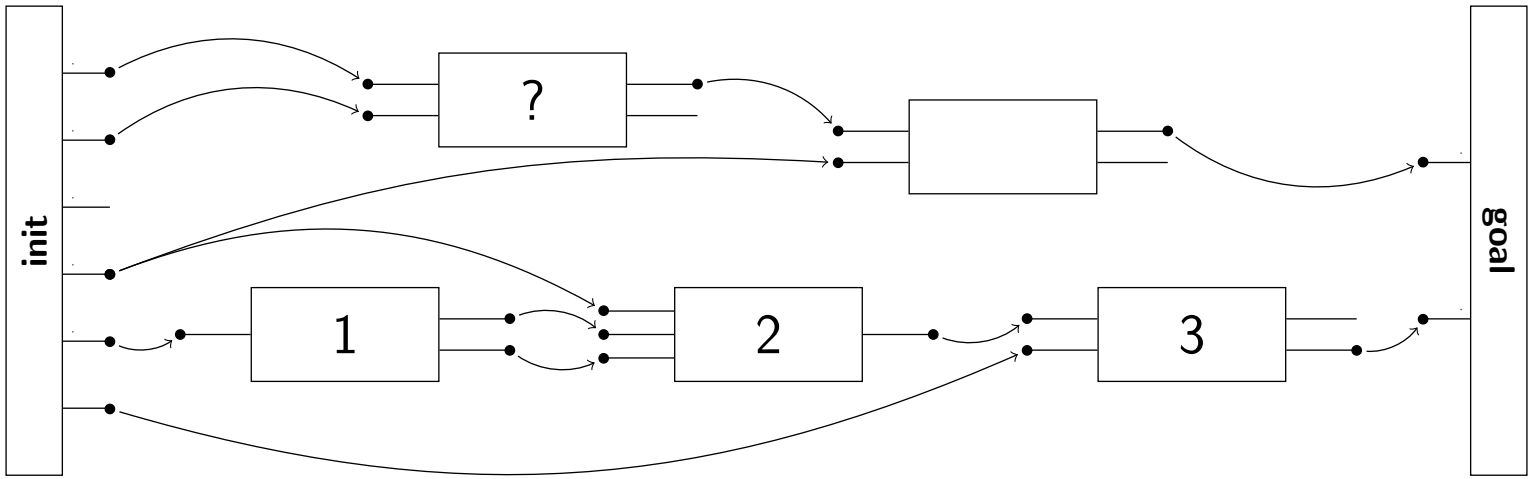
→ execute connected actions consecutively



Causal link-based plan linearization

- causal links explicitly represent the causal structure of the plan
- each link was introduced to solve a flaw – all links are required

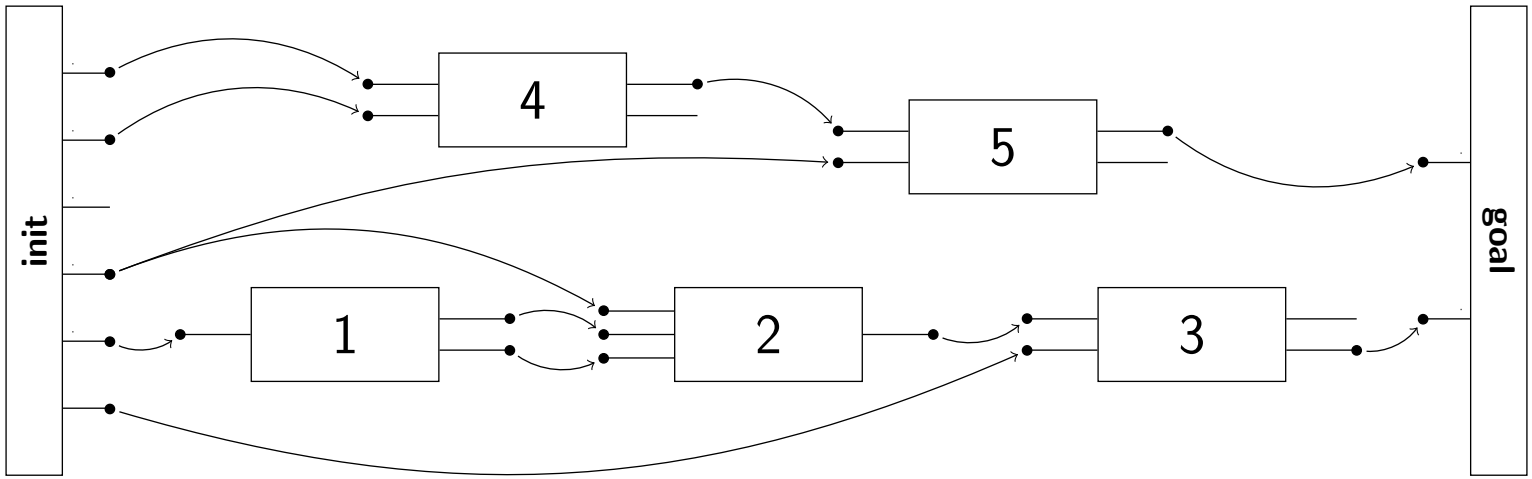
→ execute connected actions consecutively



Causal link-based plan linearization

- causal links explicitly represent the causal structure of the plan
- each link was introduced to solve a flaw – all links are required

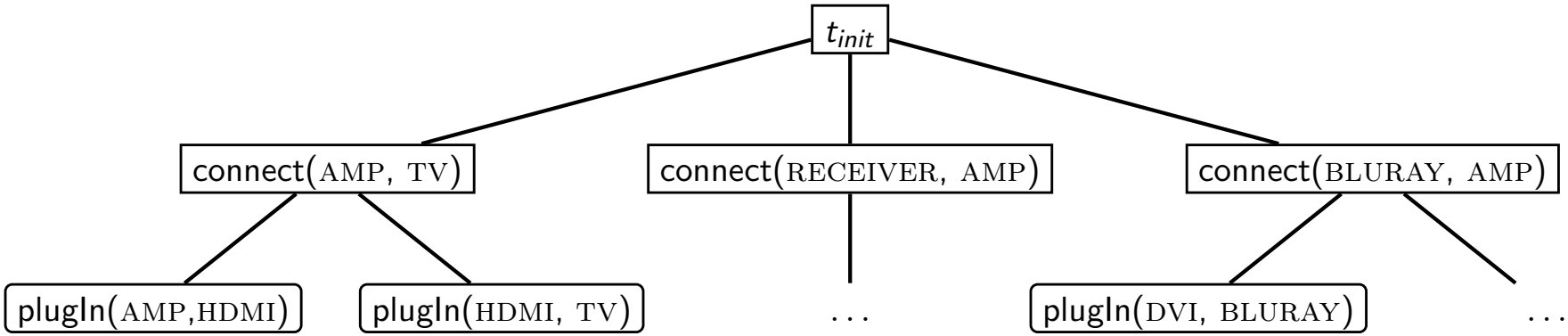
→ execute connected actions consecutively

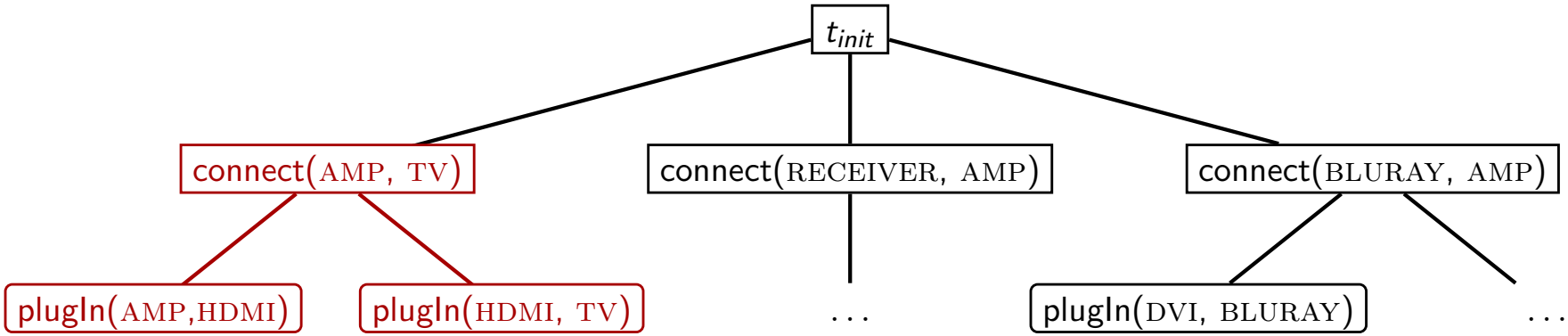


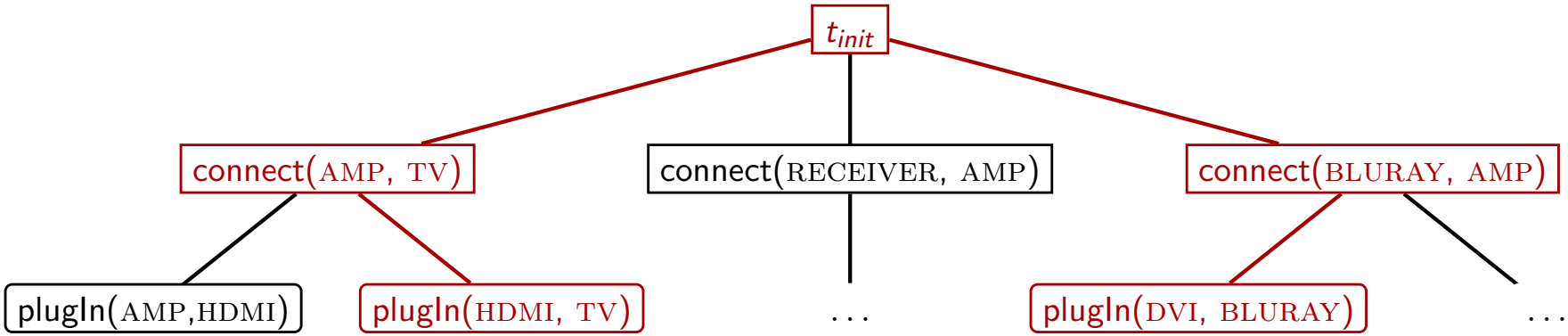
Decomposition-based plan linearization

- domain contains expert knowledge
- tasks that are introduced by the same method implement the same abstract task ($\rightarrow$ semantically related!)
- we generalize this relationship to tasks that are not in the same method ($\rightarrow$ use Task Decomposition Graph (TDG))

$\rightarrow$ execute actions that are close to each other in the TDG consecutively


t_{init}
connect(AMP, TV)
connect(RECEIVER, AMP)
connect(BLURAY, AMP)
plugIn(AMP,HDMI)
plugIn(HDMI, TV)
...
plugIn(DVI, BLURAY)
...


t_{init}
connect(AMP, TV)
connect(RECEIVER, AMP)
connect(BLURAY, AMP)
plugIn(AMP,HDMI)
plugIn(HDMI, TV)
...
plugIn(DVI, BLURAY)
...


t_{init}
connect(AMP, TV)
connect(RECEIVER, AMP)
connect(BLURAY, AMP)
plugIn(AMP,HDMI)
plugIn(HDMI, TV)
...
plugIn(DVI, BLURAY)
...

Possibilities for Empirical Evaluation

- the utilities have to be evaluated empirically
- what is the objective?
  - imitating human behavior
  - maximize subjective appraisal of humans executing a plan
  - optimize some objective (measurable) metric

Summary

- plan linearization is required if we plan for human users
- we gave three domain-independent utility functions that may help to find reasonable linearizations
- illustrated it in the “home theater” example domain
- all given utilities depend on the planning model
- causal link-based linearization additionally depends on the planning process
- outlined possible objectives and ways to evaluate the different utilities