Abstraction Planning and Diagnostics for Human-Robot Collaboration

Oliver Michael Kamperis
School of Computer Science
University of Birmingham
Primary Supervisor: Marco Castellani
Secondary Supervisor: Yongjing Wang
Nominal Supervisor: Dave Parker

Human-Robot Collaboration

The act of humans and robots operating in close proximity.

Humans may be:

1. **Cooperative**: Provide physical effort towards the goal,
2. **Commanding**: Assign goals to the agents,
3. **Exogenous**: Their actions may change the state.

Proposed evaluative HRC application domains:

- Service Robots,
- Collaborative Disassembly (HRCD),
- Collaborative Construction (HRCC).

HRCD

- Disassembly sequence planning deals with planning the sequence in which components of End-Of-Life products must be removed for reuse or recycling.
- However, diversity of products and uncertainty in the quality of their components makes it difficult.
- HRCD deals with the semi-autonomous disassembly.
- Semi-autonomous of disassembly is a middle ground in terms of cost and efficiency making it economically and environmentally beneficial.

Thesis Contributions

Aim to produce domain-independent ASP based centralised reasoning system for HRC which inherently supports the ability to reason over an arbitrary number of abstractions.

1. Support hierarchical abstraction planning and diagnostics,
2. Provide a domain-independent approach to centralised planning with teams of multiple heterogenous agents,
3. Support partial planning and oversubscription planning,
4. Enable plan-repair when plans are partially invalidated.

Abstraction in Planning

- Abstraction is integral to reasoning and problem solving.
- Low-relevancy details can be simplified away to reduce complexity or assumptions made to cope with unknowns.

The advantages of abstraction in planning are:

1. Planning in an abstract space is exponentially faster,
2. Abstract plans are much less likely to fail,
3. Abstract plans can be used to infer sub-problems in the ground space allowing the computation of partial-plans.

Planning Domain Descriptions

- Engineer first designs a planning domain description that is sufficiently expressive to deal with the application.
- Abstractions are then created through simplifications.
- Abstraction rules map state representation over abstractions.

Simplifications include:

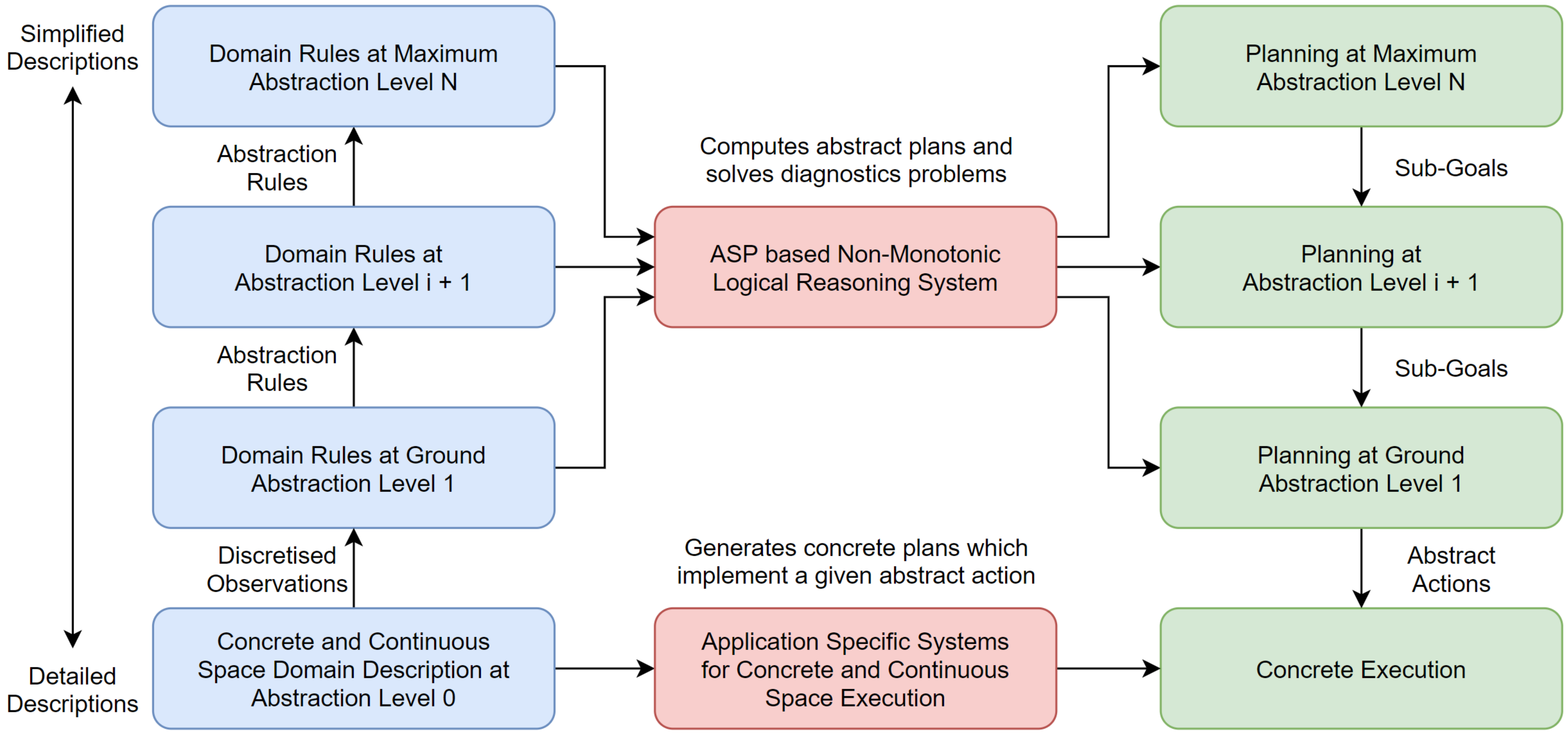
1. **Reductions**: Remove constraints which forbid actions,
2. **Refinements**: Remove detailed features of entities,
3. **Redefinitions**: Replace the original domain description with a high-level description of the given task.

Planning Domain Descriptions

A collection of logic rules at each desired level of abstraction defining the dynamic behaviour of the domain at that level.

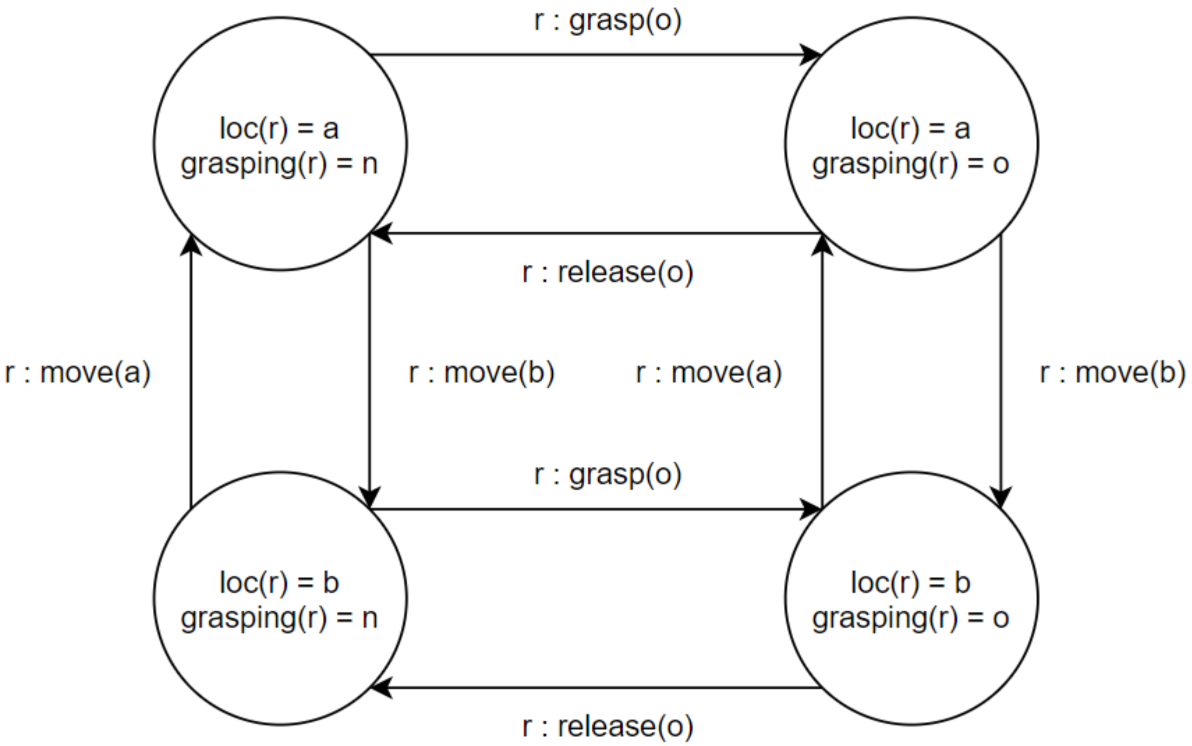
1. Action Effects
   - R : move(P) **causes** location(R) = P
2. Action Enabling Conditions (Preconditions)
   - **impossible** R : grasp(O) **if** location(R) != location(O)
3. State Constraints
   - location(O) = location(R) **if** grasping(R) = O

Planning Domain Descriptions



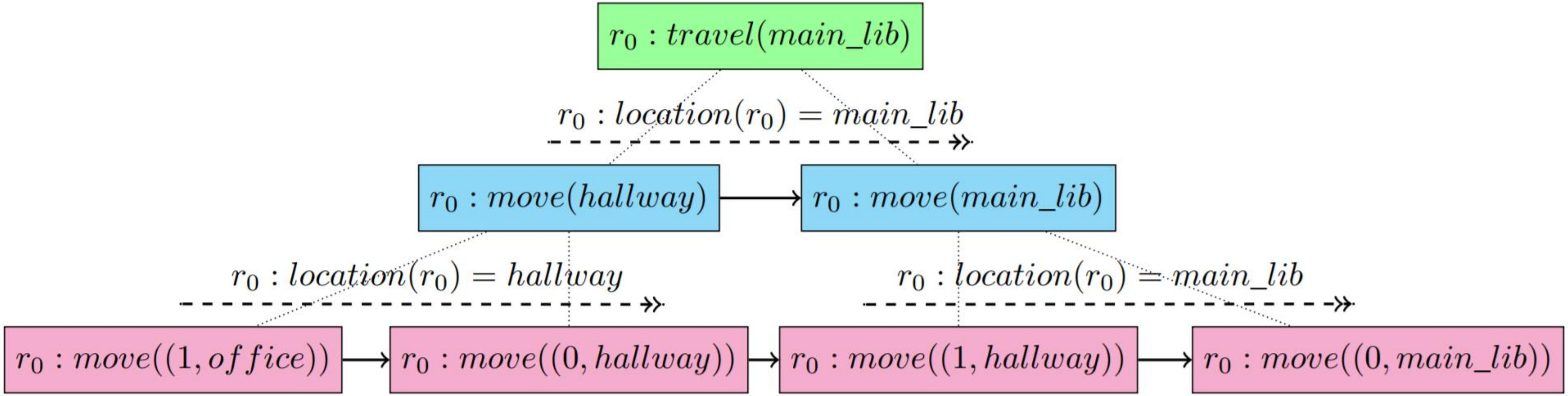
State Transition Diagrams

- Directed graph whose nodes are unique domain state and whose arcs are set of actions that transition between states.
- Planning requires finding a trajectory through that graph.



Generated Plans

- A unique trajectory is generated at each level of abstraction.
- The effects of actions planned at the higher levels are passed as sub-goals to the lower levels to ensure plan conformance.
- A plan is an ordered sequence of decomposition trees.



Decomposition Trees

- At the head is a non-empty set of actions,
- Possibly empty set of child nodes which are themselves each another discrete decomposition tree,
- Non-empty set of sub-goals achieved by its children.