Parallel Regions using a PDDL Formalization

PDDL domain as a compiler

Claudio Scheer[1]

[1]Master's Degree in Computer Science
Pontifical Catholic University of Rio Grande do Sul - PUCRS

Paper Proposal, May 2020

Table of Contents

1. Problem
2. Proposed approach
3. Results evaluation
4. Questions/Ideas
5. Schedule
6. Conclusion

Table of Contents

1. Problem
2. Proposed approach
3. Results evaluation
4. Questions/Ideas
5. Schedule
6. Conclusion

Finding parallel regions

- It takes a lot of time;
- It will cost money.

Common approaches

Static analysis of the source code:

- loops detection;

Common approaches

Static analysis of the source code:

- loops detection;
- variable dependencies;

Common approaches

Static analysis of the source code:

- loops detection;
- variable dependencies;
- identifying whether the arguments are read or written.

Table of Contents

1. Problem
2. Proposed approach
3. Results evaluation
4. Questions/Ideas
5. Schedule
6. Conclusion

PDDL domain

PDDL domain will work as a compiler.

PDDL domain

PDDL domain will work as a compiler.

- Instructions support:
  - arithmetic and binary operators;
  - functions;
  - loop.

PDDL problem

Source code will be mapped to a set of predicates.

PDDL problem

Source code will be mapped to a set of predicates.

- Goal:
  - execute all instructions;
  - must run in the correct order.

Bottleneck

Bad

Creating a set of predicates from source code will not be easy.

Good

It can be easily automated.

Table of Contents

1. Problem
2. Proposed approach
3. Results evaluation
4. Questions/Ideas
5. Schedule
6. Conclusion

Results evaluation

- Decode planner output to source code;
- Validate the parallel execution;
- Was the execution time shorter?
  - I should probably test a big problem;
  - I may not have enough time.

Table of Contents

1. Problem
2. Proposed approach
3. Results evaluation
4. Questions/Ideas
5. Schedule
6. Conclusion

Questions/Ideas

1. Is the compiler domain capable of handling fluent variables and predicates?

Questions/Ideas

1. Is the compiler domain capable of handling fluent variables and predicates?
2. Is the compiler domain capable of performing operations with strings?

Questions/Ideas

1. Is the compiler domain capable of handling fluent variables and predicates?
2. Is the compiler domain capable of performing operations with strings?
3. Which planners should I test the compiler domain on?

Questions/Ideas

1. Is the compiler domain capable of handling fluent variables and predicates?
2. Is the compiler domain capable of performing operations with strings?
3. Which planners should I test the compiler domain on?
4. How does a planner find a parallel region?

Questions/Ideas

1. Is the compiler domain capable of handling fluent variables and predicates?
2. Is the compiler domain capable of performing operations with strings?
3. Which planners should I test the compiler domain on?
4. How does a planner find a parallel region?
5. Can I set a weight for the planner to get regions that are really worth running in parallel?

Table of Contents

1. Problem
2. Proposed approach
3. Results evaluation
4. Questions/Ideas
5. Schedule
6. Conclusion

Schedule

Task	**Start**	**End**
Understand better compilers	06-01-2020	06-03-2020
Support sum instruction	06-03-2020	06-07-2020
Support proposed instructions	06-08-2020	06-15-2020
Evaluate results	06-16-2020	06-20-2020
Write paper	06-20-2020	06-25-2020

Table of Contents

1. Problem
2. Proposed approach
3. Results evaluation
4. Questions/Ideas
5. Schedule
6. Conclusion

Conclusion

- This is not a conventional approach;

Conclusion

- This is not a conventional approach;
- If the results are positives, the approach may reduce the amount of time to find parallel regions.