An Application for a Certified Grid Computing Framework

Parallel Theorem Proving for Linear Logic

Bor-Yuh Evan Chang

Advisors: Prof. Robert Harper and Prof. Frank Pfenning
December 12, 2001
Suppose you had an ingeniously crafted massively parallelized algorithm to solve some problem. You would like use all the “wasted” computing resources of the Internet.

**Problem**: How does a resource *donator* know you are a benevolent researcher and not a evil hacker?
The Big Picture – the ConCert Project

- The ConCert project proposes to use *certified code* to resolve this issue of trust.

**Vision:** Distributed-application developer’s utilization of donated resources is completely transparent to the donator, but the donator is confident the specified safety, security, and privacy policies will not be violated.
My Contribution

**Idea:** The process of developing a substantial application using the ConCert infrastructure will help us better understand the requirements on the infrastructure and how to program in such an environment.

- **Goals**
  - Make apparent the current shortcomings
  - Drive the architecture to a more robust and stable state
  - Better understand the requirements from a programmer’s perspective

- **What Application?**
  - A *bottom-up* parallel theorem prover for intuitionistic linear logic
    - **Advantages**
      - The *focusing* strategy helps with producing independent subproblems
      - Able to check validity of results easily
      - Few existing linear logic provers
    - **Concerns**
      - How to balance the cost of communication
      - How to limit frivolous parallelism
Parallelism in Theorem Proving

- **AND-parallelism**

- **OR-parallelism** ← exploitable
Algorithm Overview

- Focusing [Andreoli ’92][Pfenning ’01]
  - Refinement of the plain sequent calculus to reduce the non-determinism in proof search
  - Advantageous for parallelization by concentrating several non-deterministic choices into one place
  - Procedure:
    - first apply invertible rules eagerly
    - select a “focus” proposition and apply non-invertible rules until reach an atom or an invertible connective
    - upon reaching an atom, proof attempt either fails or succeeds
Resource-distribution via Boolean constraints [Harland and Pym ’01]

- Method to postpone the distribution of resources for multiplicative connectives

\[ \Gamma; \Delta_1 \Rightarrow A \quad \Gamma; \Delta_2 \Rightarrow B \]

\[ \Gamma; (\Delta_1, \Delta_2) \Rightarrow A \otimes B \quad \otimes \mathbb{R} \]

- Represent constraints using OBDDs (Ordered Binary Decision Diagrams)
Focusing

Sequential Implementation

Concurrent Implementation
Current Status

- Built a working non-concurrent prover in SML
- Modified prover to introduce concurrency using CML

Next Steps

- Theorem Proving Optimizations
  - eliminate spurious focusing based on logical compilation
  - integrate more efficient OBDD implementation
- Extend theorem prover to return proofs
- Integrate with the ConCert infrastructure