An Application for a Certified Grid Computing Framework

Bor-Yuh Evan Chang

Advisors: Professors Robert Harper and Frank Pfenning

Carnegie Mellon University
October 31, 2001
The Big Picture - ConCert

Resource Donators
**The Big Picture - ConCert**

**Vision:** Distributed-application developer 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.
ConCert Framework - Conductor

- Joshua Dunfield
  - basic protocol and system for distributing and verifying software
  - makeshift certifying Standard ML compiler
- Margaret DeLap
  - examining load balancing and task brokering issues
My Research Plan

- Develop a substantial application using the ConCert framework and make the ConCert framework capable of supporting such an application

- Goals
  
  - make apparent the current shortcomings
  
  - drive the architecture to a more robust and stable state
  
  - work on the framework top-down
What Application?

- Parallel Theorem Prover

Why?

- Check validity of results easily
- Build upon my previous experience
What’s Happening

- Investigated adapting existing theorem provers (Gandalf, E)

- Decided to develop our own - a subgoal-reduction based parallel theorem prover for intuitionistic linear logic
  
  - Advantages:
    
    * focusing strategy helps with independent subproblems
    * few existing linear logic provers
  
  - Concerns:
    
    * uncertain about cost of communication
Utilizing the ConCert Framework

• Parallelism in theorem proving
  – AND-parallelism
  – OR-parallelism \( \leftarrow \) exploitable

• Conductor requirements
  – program can specify new thread on this machine or another machine
  – framework manages how thread is distributed
  – program can signal thread to terminate
How do we write code for Conductor?

**Ideal:**

\[(fn() => ... a ... b ...)]

**Currently:**

\[(fn(a,b) => ... )\]
Next Steps

1. Complete an implementation of the theorem prover in CML
2. Develop mechanism to communicate with the framework to spawn a thread on another machine
3. Factor out functions to spawn
4. Develop means for the developer to kill threads on other machines