

Iktara in ConCert

Realizing a Certified Grid Computing
Framework from a Programmer's Perspective

BoF Yuh Evan Chang

May 8, 2002

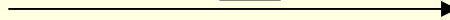
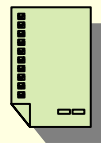
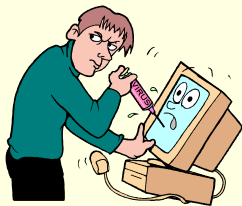
Advisors: Prof. Robert Harper and Prof. Frank Pfenning

ConCert

- Suppose you have 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 donor* know you are a benevolent researcher and not an evil hacker?



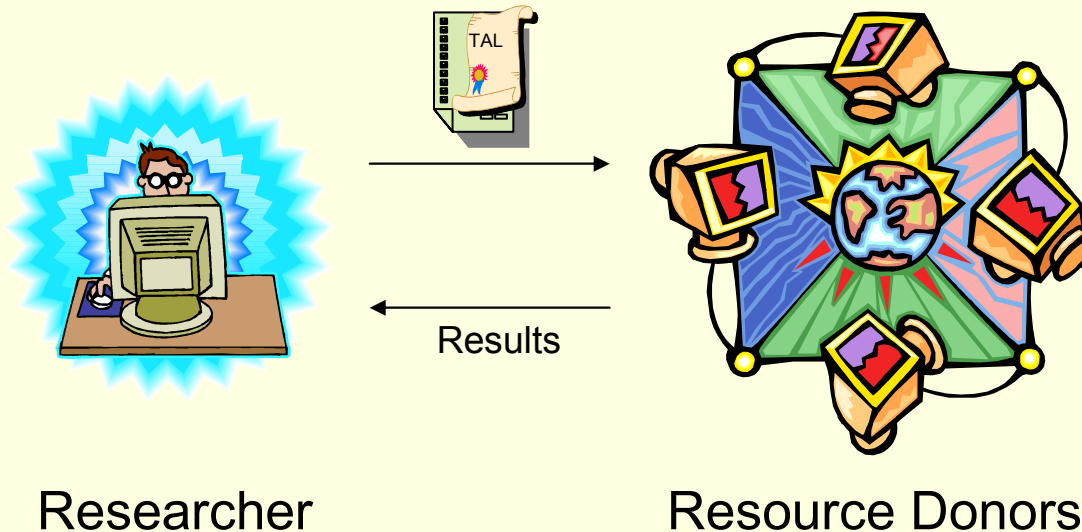
OR



Resource Donors

ConCert

- 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 donor, but the donor 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 framework will help us better understand the requirements on the framework and how to program in such an environment.

■ Goals

- Make apparent the current shortcomings.
- Drive the framework to a more robust and stable state.
- Better understand the requirements from a programmer's perspective.
- Design a programming model based on these observations.

■ What Application?

- A *bottom-up* parallel theorem prover for intuitionistic linear logic (Iktara)
 - 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

$$\frac{\Gamma; \Delta \Longrightarrow A \quad \Gamma; \Delta \Longrightarrow B}{\Gamma; \Delta \Longrightarrow A \ \& \ B} \ \&R$$



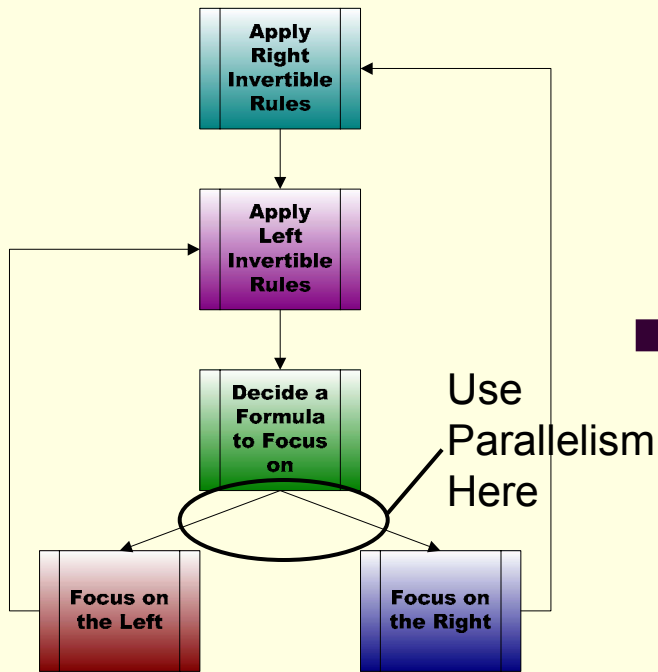
■ OR-parallelism

$$\frac{\Gamma; \Delta \Longrightarrow A}{\Gamma; \Delta \Longrightarrow A \ \oplus \ B} \ \oplus R_1$$

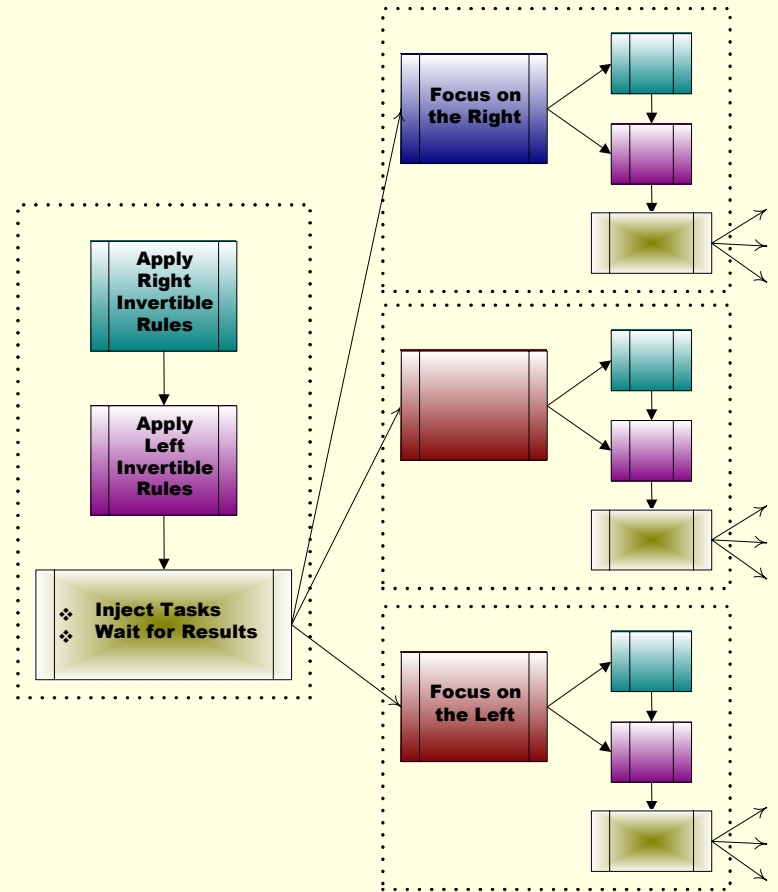
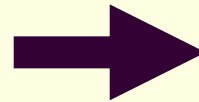
$$\frac{\Gamma; \Delta \Longrightarrow B}{\Gamma; \Delta \Longrightarrow A \ \oplus \ B} \ \oplus R_2$$



Focusing

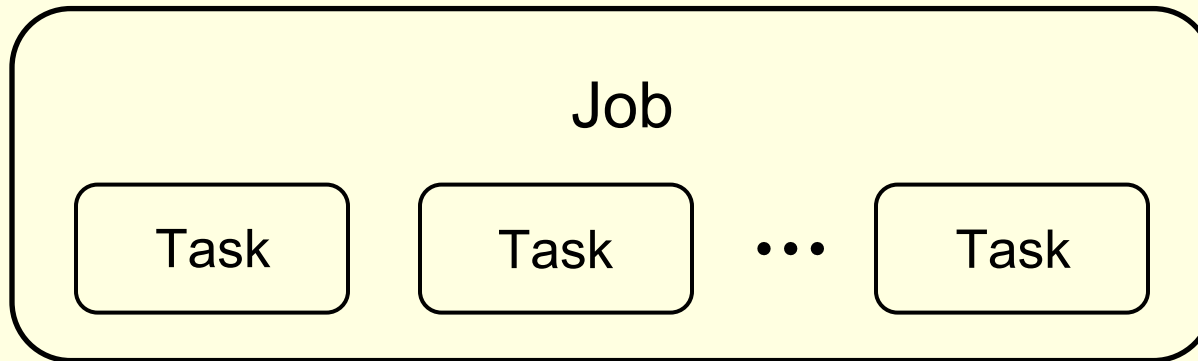


Sequential Implementation



Parallel Implementation

Jobs and Tasks



■ *Job*

- A whole program
- Injected into the network from the command-line
- Unit of computation from the grid-application user's point of view

■ *Task*

- Unit of computation from the programmer's point of view
- Consists of a piece of closed code along with its arguments

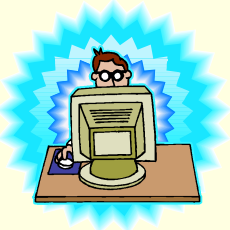
Failure



- Tasks should be restartable and each run is as “good” as any other
- Tasks communicate only through sending and receiving of results
- Programs should be kept until result has been computed

■ Problem:


- There are multiple “ways” to prove some sub-goals.
- The “way” a sub-goal is proven may affect the provability of other sub-goals.
- Need communication?






Multiple Results

■ Solution:

- 
- Have each sub task return “all possible” results.
 - More specifically, each sub task returns a stream of results.

■ Programming Support:

- 
- Return code as part of the result that represents “what to do next (if needed).”
 - Have the ability to “register” code on the network without starting the computation.



Future Work

■ Iktara - Theorem Prover

- Integrate with ConCert software

■ Programming Model

- Implement compiler
- Find how to determine if or ensure that data is *marshalable*
- Garbage collect tasks?
- More primitive constructs?





More Information

- ConCert Project Webpage
 - <http://www.cs.cmu.edu/~concert>