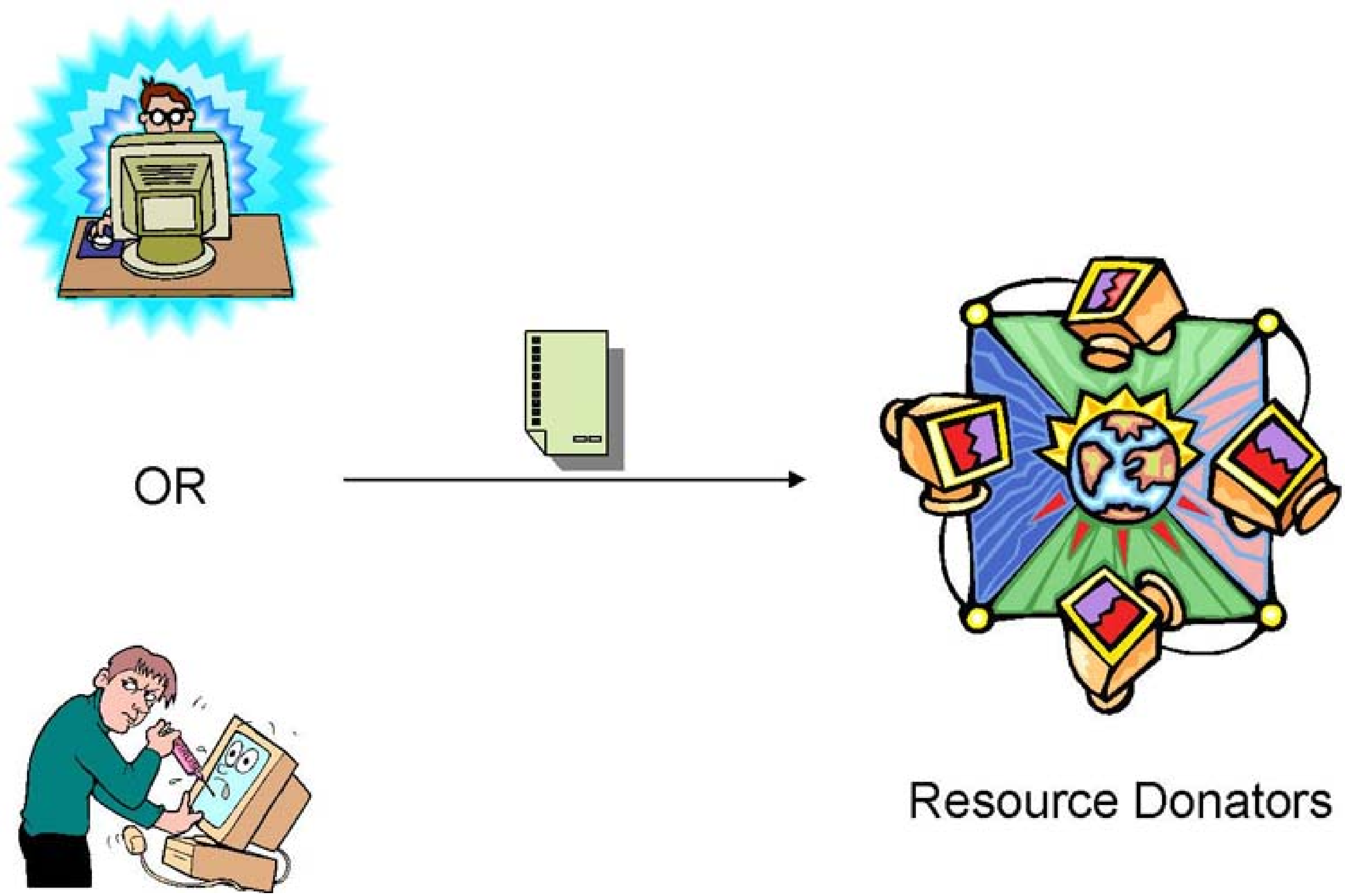
Parallel Theorem Proving for Linear Logic

Bor-Yuh Evan Chang

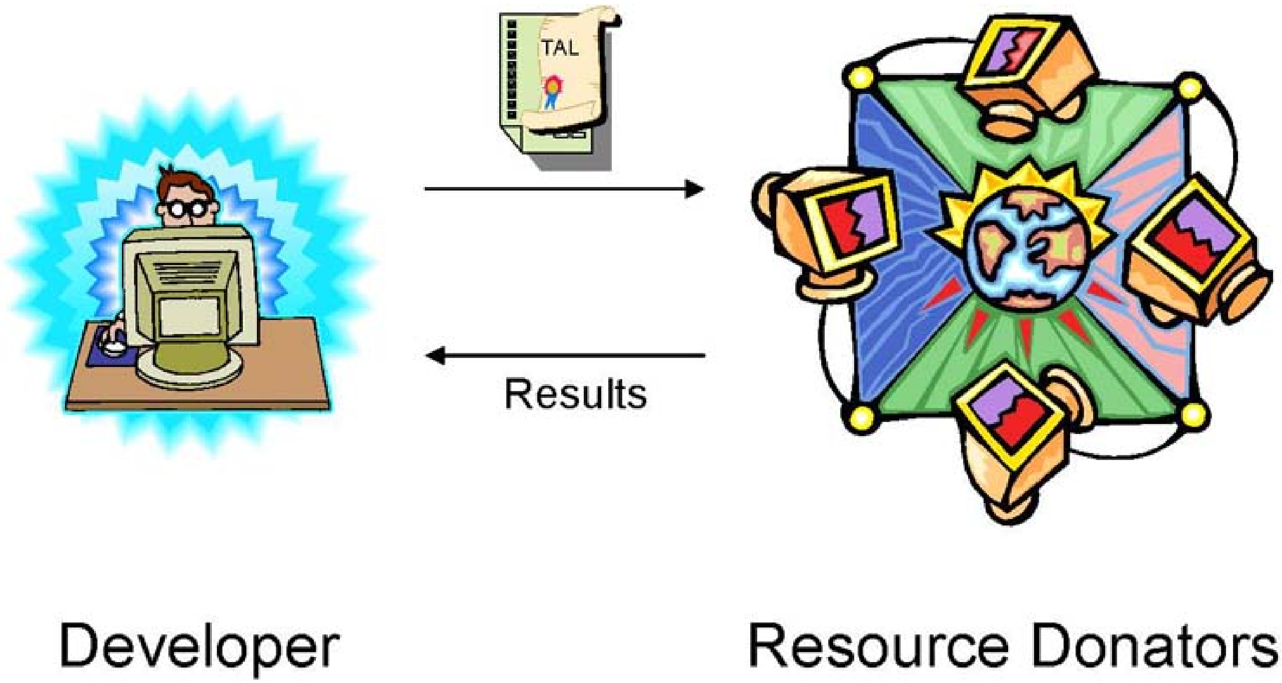
Advisors: Professors Robert Harper and Frank Pfenning

ConCert Project Meeting
Carnegie Mellon University
December 10, 2001

ConCert Vision



ConCert Vision



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.

Why Theorem Proving?

Idea: The process of developing a parallel theorem prover 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 infrastructure to a more robust and stable state
  - work on the infrastructure top-down

Approach

- Develop a subgoal-reduction based parallel theorem prover for intuitionistic linear logic
  - Advantages:
    - *focusing* strategy helps with 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

Current Plan

1. Build a working non-concurrent prover in SML. $\surd$
2. Modify prover to introduce concurrency using CML. $\surd$
3. Understand the (communication) requirements on the infrastructure and where refinements should be made.
4. Tie in with Margaret's work on the infrastructure.

Parallelism in Theorem Proving

- AND-parallelism

$$\dfrac{\begin{array}{c}\vdots \\ \Gamma ; \Delta \Longrightarrow A\end{array} \qquad \begin{array}{c}\vdots \\ \Gamma ; \Delta \Longrightarrow B\end{array}}{\Gamma ; \Delta \Longrightarrow A \,\&\, B} \;\&\mathsf{R}$$

- OR-parallelism $\leftarrow$ exploitable

$$\dfrac{\begin{array}{c}\vdots \\ \Gamma ; \Delta \Longrightarrow A\end{array}}{\Gamma ; \Delta \Longrightarrow A \oplus B} \;\oplus\mathsf{R}_1 \qquad \dfrac{\begin{array}{c}\vdots \\ \Gamma ; \Delta \Longrightarrow B\end{array}}{\Gamma ; \Delta \Longrightarrow A \oplus B} \;\oplus\mathsf{R}_2$$

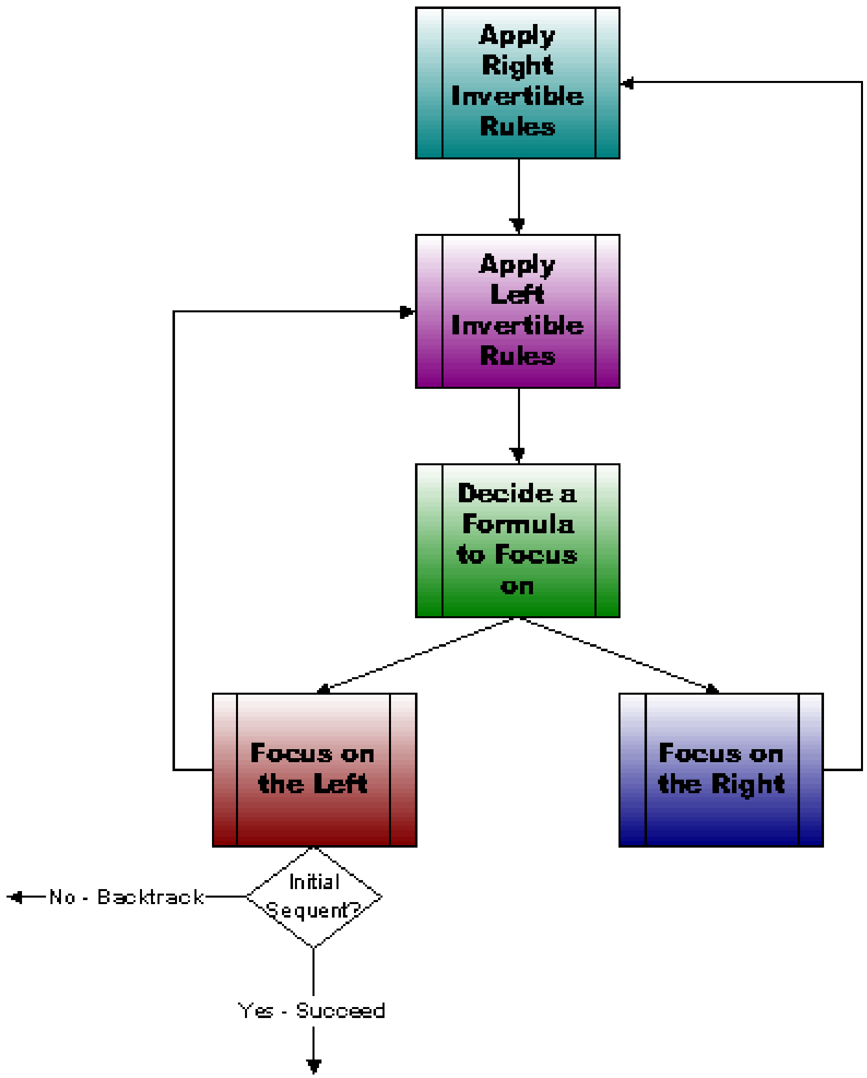
Core Algorithm

- Focusing Strategy [Andreoli '92][Pfenning '01]
  - first apply invertible eagerly
  - select a "focus" proposition and apply non-invertible rules until reach invertible or atomic
- *Resource-distribution via Boolean constraints* [Harland and Pym '01]

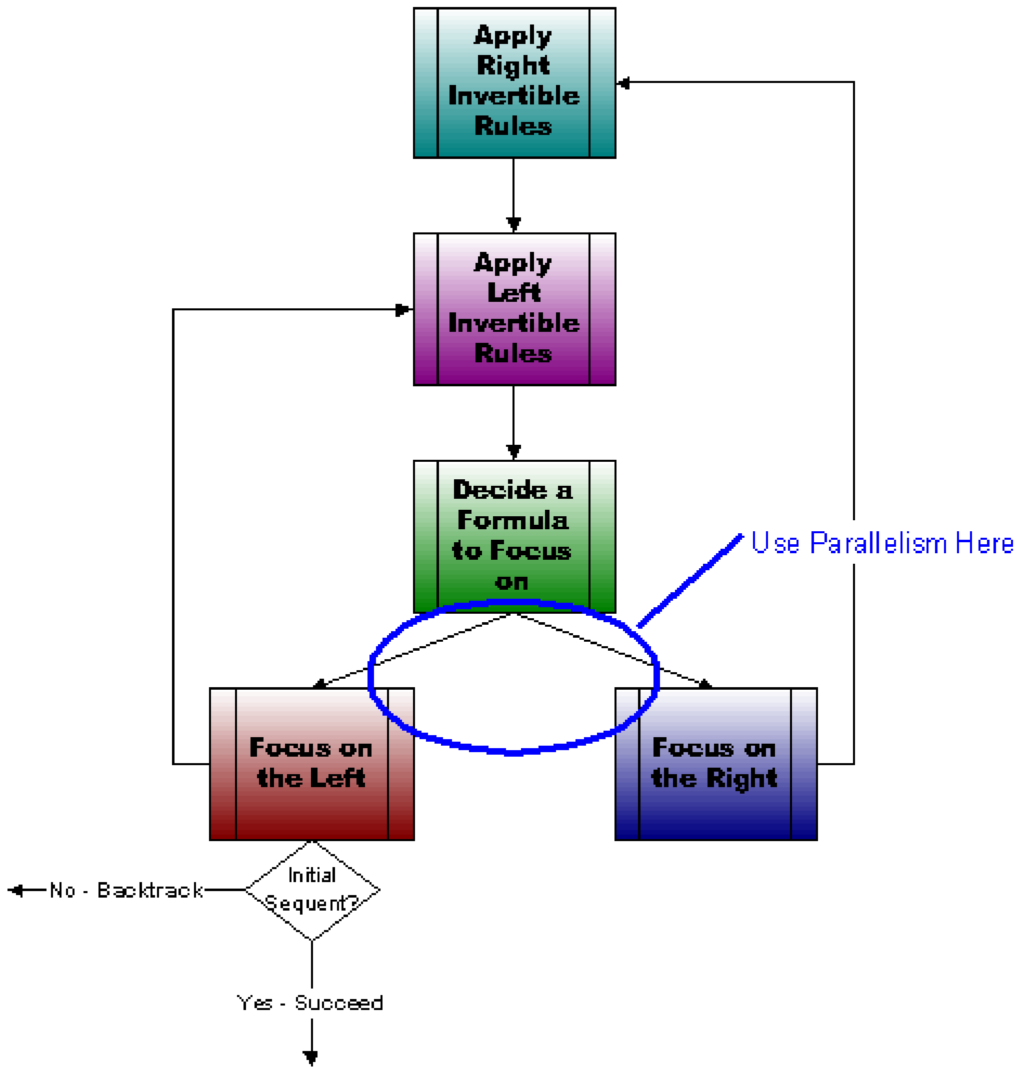
$$\dfrac{\begin{array}{c}\vdots \\ \Gamma; \Delta_1 \Longrightarrow A\end{array} \qquad \begin{array}{c}\vdots \\ \Gamma; \Delta_2 \Longrightarrow B\end{array}}{\Gamma; (\Delta_1, \Delta_2) \Longrightarrow A \otimes B} \otimes\mathsf{R}$$

  - represent constraints using OBDDs

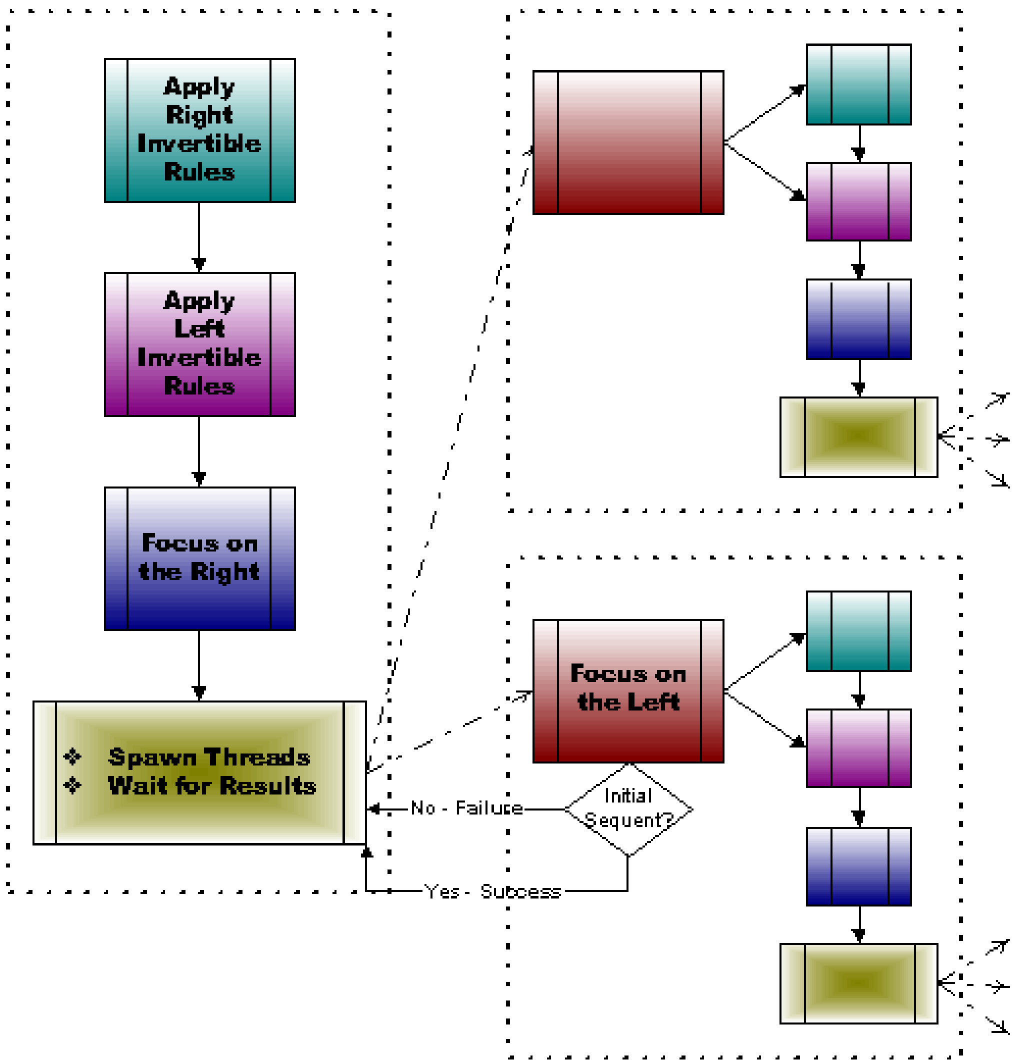
Focusing (Sequential)



Focusing (Sequential)

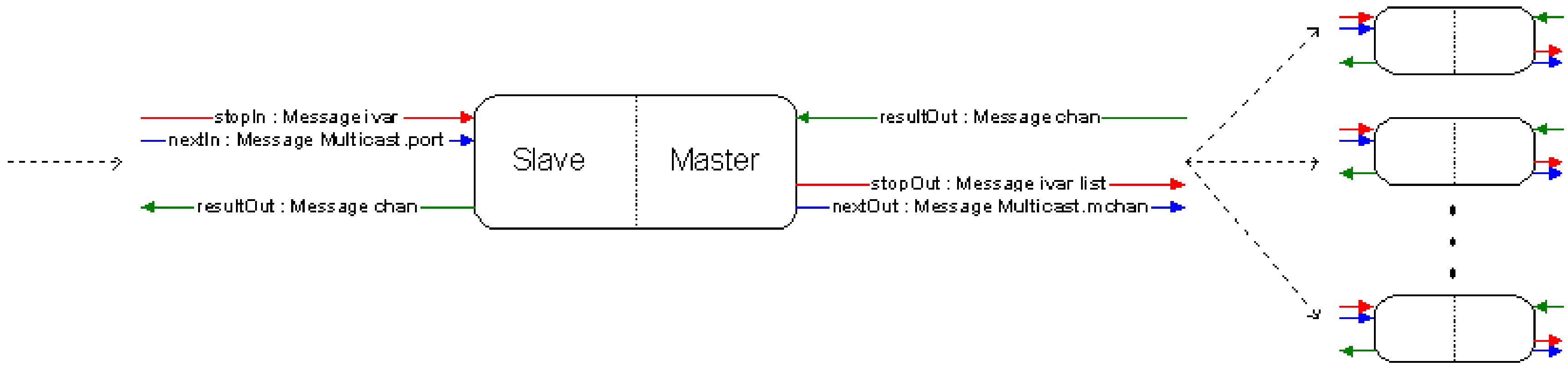


Focusing (Concurrent)

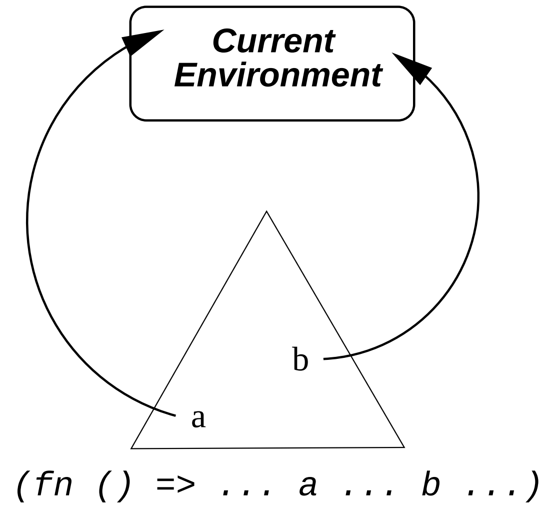


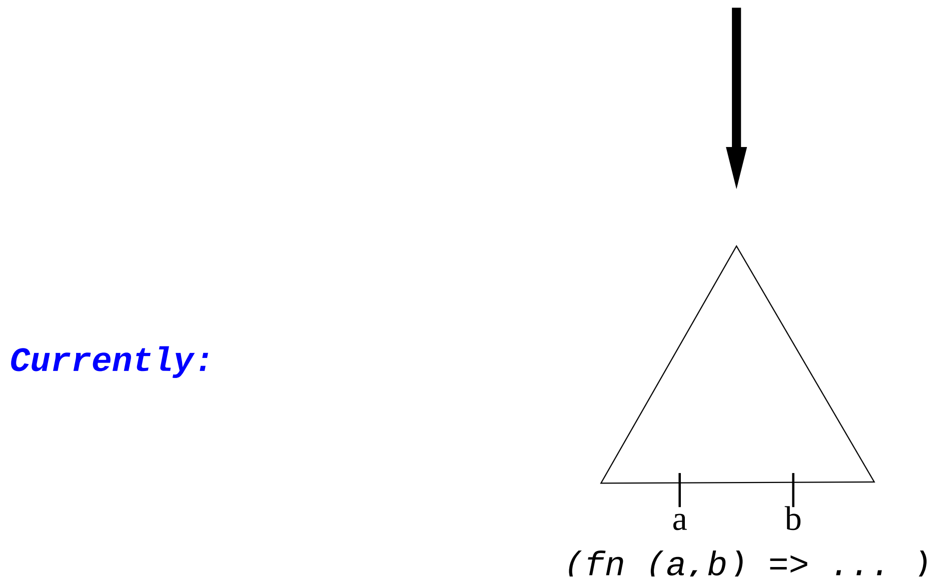
Communication (CML)

Message	::=	Failure(thread_id)
	|	Success(constraints)
	|	STOP
	|	NEXT



Integrating into the ConCert Infrastructure





Summary of Requirements on Infrastructure

- program can specify new thread on this machine or another machine
- framework manages how thread is distributed
- basic communication mechanism (to pass STOP or NEXT signals)

Next Steps

1. Theorem Proving Optimizations

   (a) Eliminate spurious focusing

   (b) Integrate more efficient OBDD implementation

2. Extend theorem prover to return proofs

3. Integrate with the ConCert infrastructure

DEMO