Peer-to-Peer Infrastructure - Branch & Bound API

Alexandre di Costanzo

The 2nd ProActive User Group

Monday October 10th 2005

Outline

- Peer-to-Peer Infrastructure
  - Description
  - Experimentations
- Branch & Bound API
  - Description
  - Experimentations
- Future and On Going Work
- Conclusion

Peer-to-Peer Infrastructure

Self-Organized and Configurable

Motivations and Goals

- Using spare CPU cycles of desktop machines:
  - Host not available all the time
  - Used by their normal users
- Providing a permanent shared JVMs network for computing
- Not a new communication protocol, not a DHT
- Self-Organized and Configurable

The P2P Infrastructure

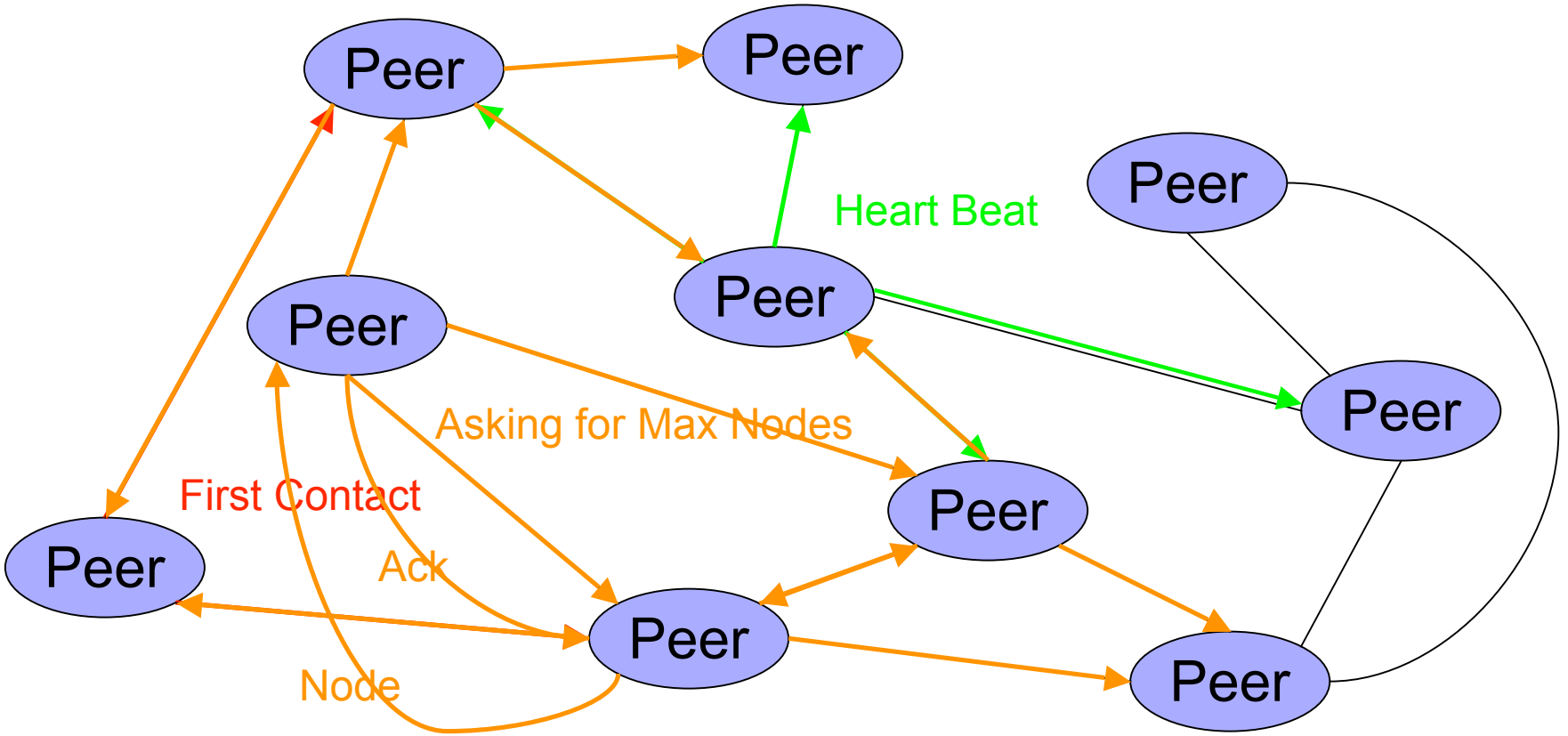
- Dynamic environment:
  - Bootstrapping (First contact)
  - Discovering peers
  - Acquiring Computational nodes
- Self Organized and Configurable:
  - Time To Update (TTU): peers availability
  - Number Of Acquaintances (NOA): keep the infrastructure up
  - Time To Live (TTL): in host hop for message life
  - First Gnutella message protocol version inspired

A Gnutella Inspired Protocol

Breadth-First Search algorithm (BFS)

- **Sending** a message with an **UID**, and **TTL,** and **number of asked nodes,** and **service reference**
- **Receiving**
  - **Is it an old message?**
    - **Yes**, it is: continue;
    - **No**, it‘s not:
      - Keep the UID
      - I have a **free node**:
        - **Send** the node reference to the callee and waiting an **ACK** until timeout
      - If **timeout** is reached or **NACK**
        - continue;
      - If **ACK** and **asked nodes - 1 > 0** and **TTL > 0** then
        - **Broadcast** with **TTL - 1** and **asked nodes - 1**
      - continue;

P2P Infrastructure



NOA = 2; TTU = 1 minutes; TTL = 2

N-Queens With n = 25

INRIA P2P Desktop GRID
6 months of computation

- Total of solutions found:
  2,207,893,435,808,352 ≈ **2 quadrillions**
- Total of tasks computed: **121,251,992**
- Average time of one task computation:
  2 minutes and 18 seconds
- Total computation time ≈ **185 days**
- Total workers CPU time ≈ **53 years**
- Total of unique machines: **260**

Branch and Bound API

Dynamic and Simple

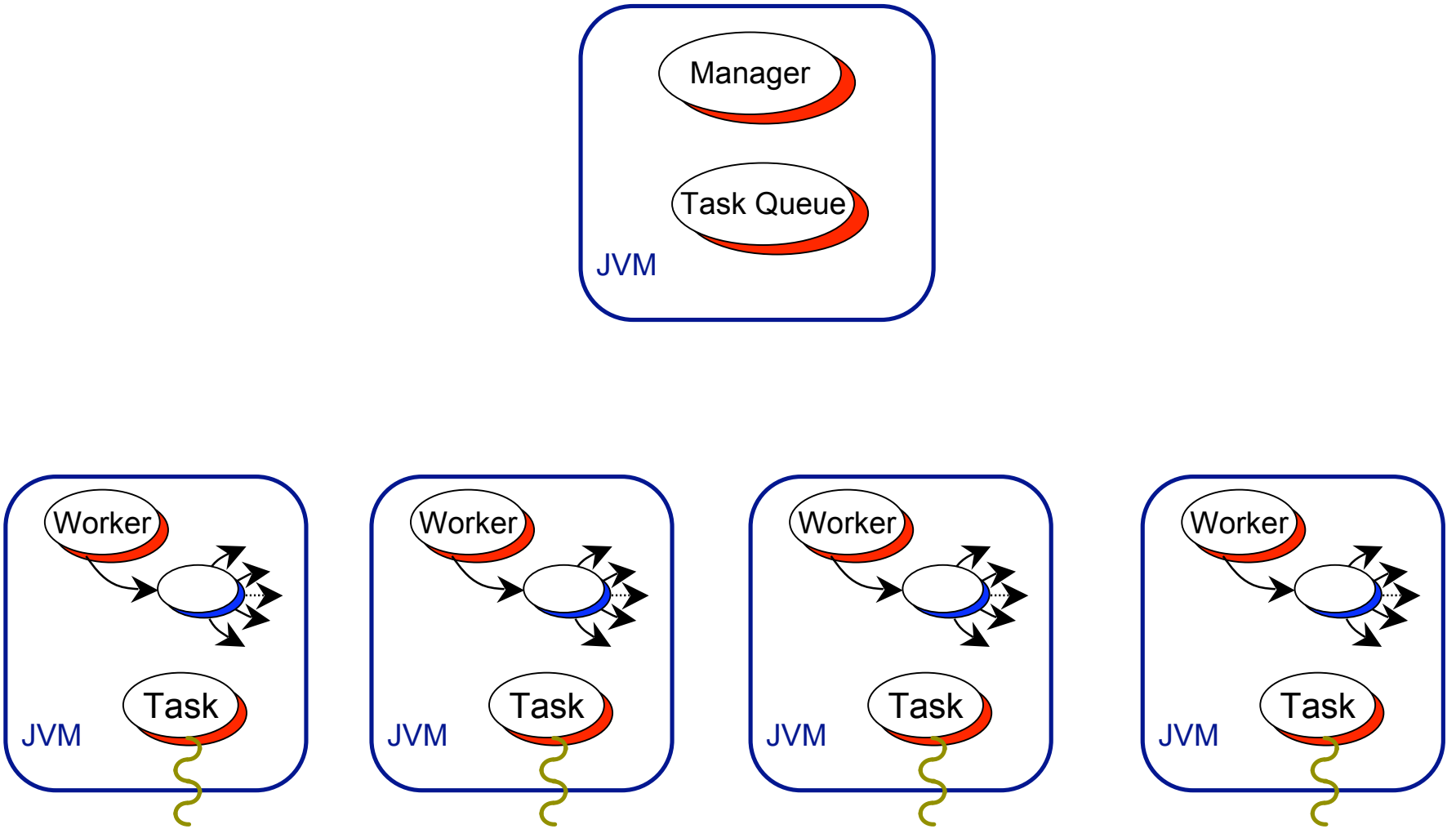
Branch & Bound API (BnB)

- Provide a high level programming model for solving BnB problems, which manages task distribution and provides task communications
- Goals:
  - Exploring a search tree in parallel with task communications for cutting bad tree branches
  - For the user the program distribution is hidden
  - Based on the Farm Skeleton (Bag of Task)
  - **NOT** only for P2P
- Features:
  - Dynamic task split
  - Automatic result gather
  - Broadcasting best current result
  - Automatic backup and task reallocation (configurable)
  - Choose and/or Create the queue for task allocation.

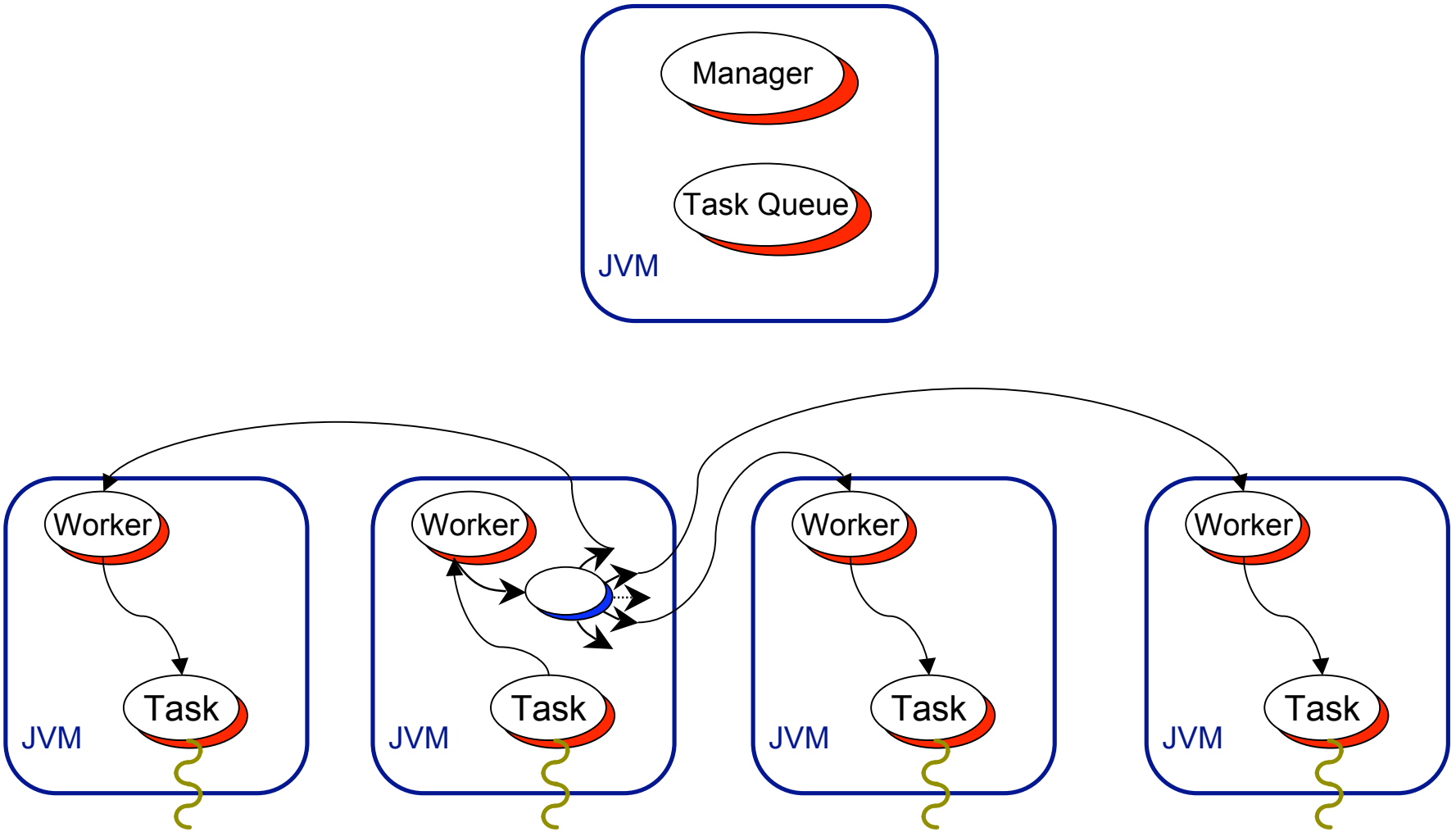
BnB: Task Queue Interface

- Providing tasks, and managing results
- Backup current solutions and none achieved tasks
- For the moment:
  - Basic Queue: FIFO
  - Larger Queue: Explore the search tree in larger
- User can implements is own Task Queue

Global Architecture



Broadcasting a New Best Solution



BnB API

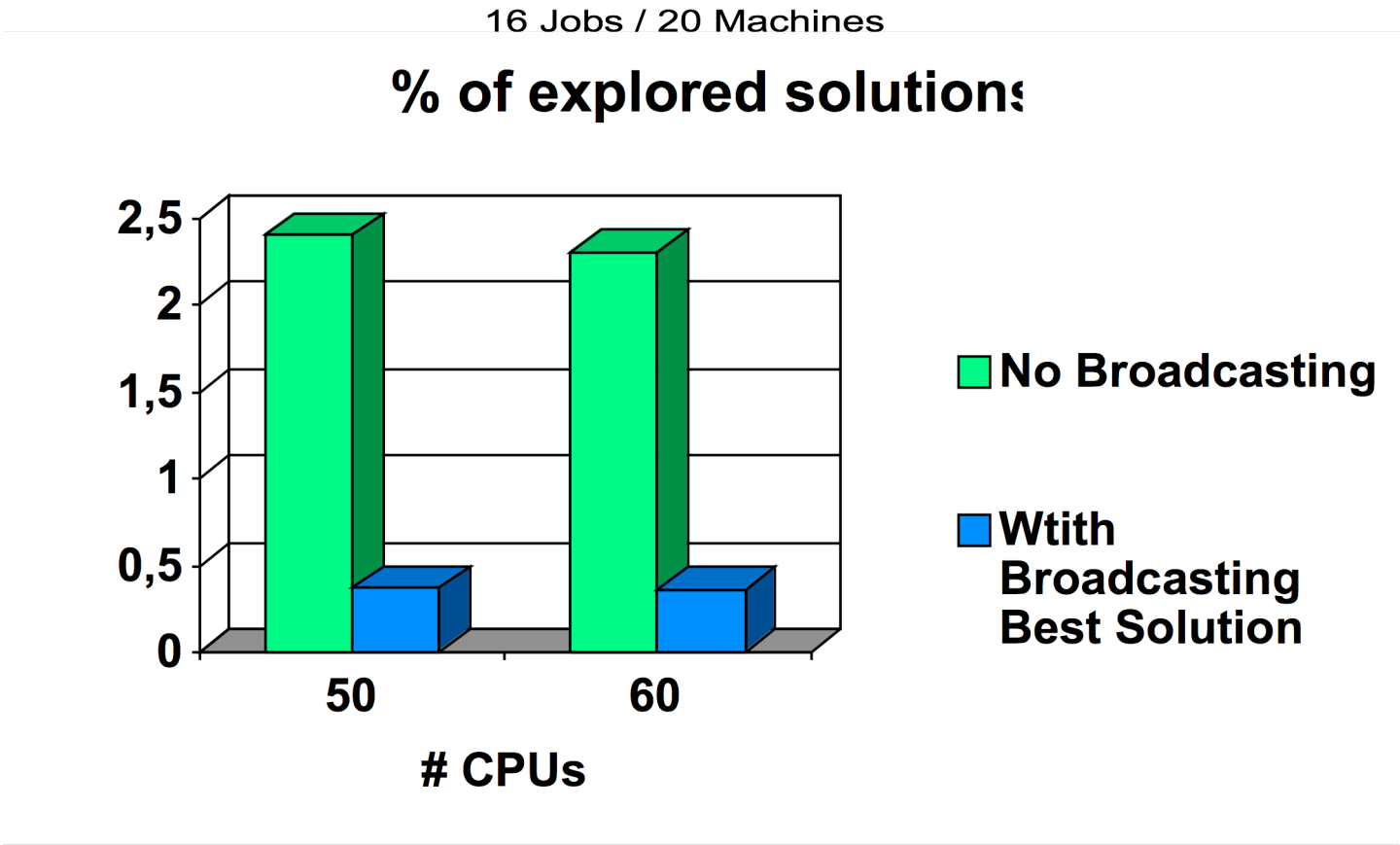
Just extend the abstract Task:

```
package org.objectweb.proactive.branchnbound.core;

public abstract class Task implements Serializable,
Comparable {
    protected Worker worker;
    protected Object bestKnownResult;

    public abstract Result execute();
    public abstract Vector split();
    public Result gather(Result[] results) {
        // This call is asynchronous
        // Have a default behavior
    }

    public abstract void initLowerBound();

    public abstract void initUpperBound();

    public int compareTo(Object arg) {
        // Have a default behavior
    }

}
```

And start the computation:

```
import org.objectweb.proactive.branchandbound.*;
import org.objectweb.proactive.branchendbound.core.*;

Manager manager = ProActiveBranchNBound.newFarmWithBasicQueue(myTask, virtualNode);
Result futureResult = manager.start();
```

FlowShop Experimentations



Cluster of 32 bi-Opteron @ 2Ghz, connected via Gigabit Ethernet

On Going and Future Work

- P2P Infrastructure:
  - INRIA Coprin Research Group:
    - Running Alias library:
      - Kochen-Specker
      - 3 years of cumulated time since the July 22th 2005
  - General improvements
- Branch and Bound API:
  - Auto-dynamic splitting
  - Providing more Queues
  - More experimentations
  - Wrapping native code

Conclusion

- A Self-Organized P2P Infrastructure for providing JVMs:
  - Deployed and used at INRIA Sophia
- A simple API for distributing and solving Branch & Bound problem:
  - Hiding distribution
  - Open API

Further Information

- The P2P:

  Since ProActive 2.2

- The Branch and Bound API:

  ProActive 3.0

- On Tuesday October 11th:

  ProActive Tutorial Hands-On Grid Programming

  P2P Demo with the INRIA P2P Desktop Grid