Universität Stuttgart

Institute of Parallel and
Distributed Systems (IPVS)

Universitätsstraße 38
D-70569 Stuttgart

Direct Backtracking: An Advanced Adaptation Algorithm for Pervasive Applications

Stephan Schuhmann, Klaus Herrmann, Kurt Rothermel

Institute of Parallel and Distributed Systems (IPVS)

Universität Stuttgart, Germany

Email: *{firstname.lastname}@ipvs.uni-stuttgart.de*

Outline

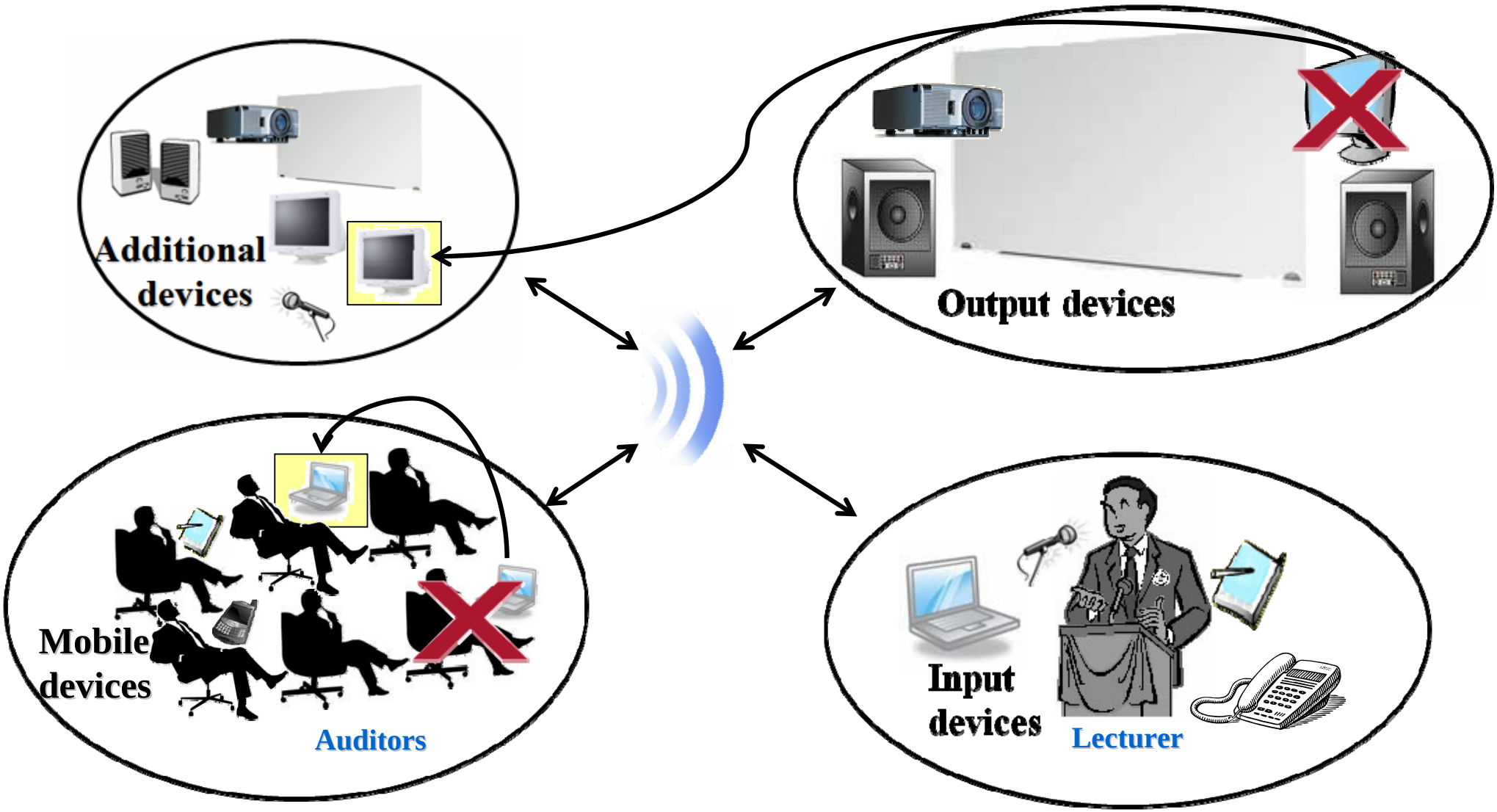
- Motivation
- System Model
- Related Work
- The Direct Backtracking Algorithm
  - General view
  - Proactive Backtracking Avoidance
  - Intelligent Backtracking
- Evaluation
- Conclusion

Motivation

- Applications for Pervasive Computing
  - Pervasive Applications in focus of many current projects
  - Due to dynamic nature, configuration of applications necessary before execution
  - Decentralized configuration mandatory in ad hoc scenarios
  - *But:* Most realistic scenarios are heterogeneous
    - Centralized configuration can reduce latencies and communication overhead
- Adaptations before and during runtime necessary because of
  - limited and scarce resources
  - changing environmental conditions

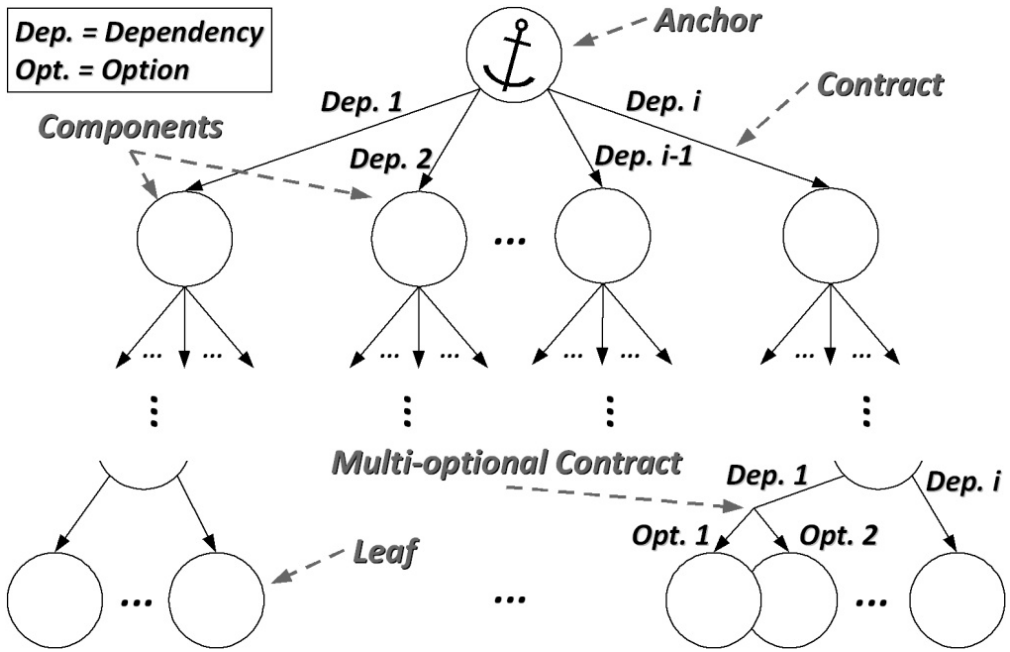
⇒ Efficient and resource-conserving configuration and adaptation desirable

Pervasive Computing Scenario

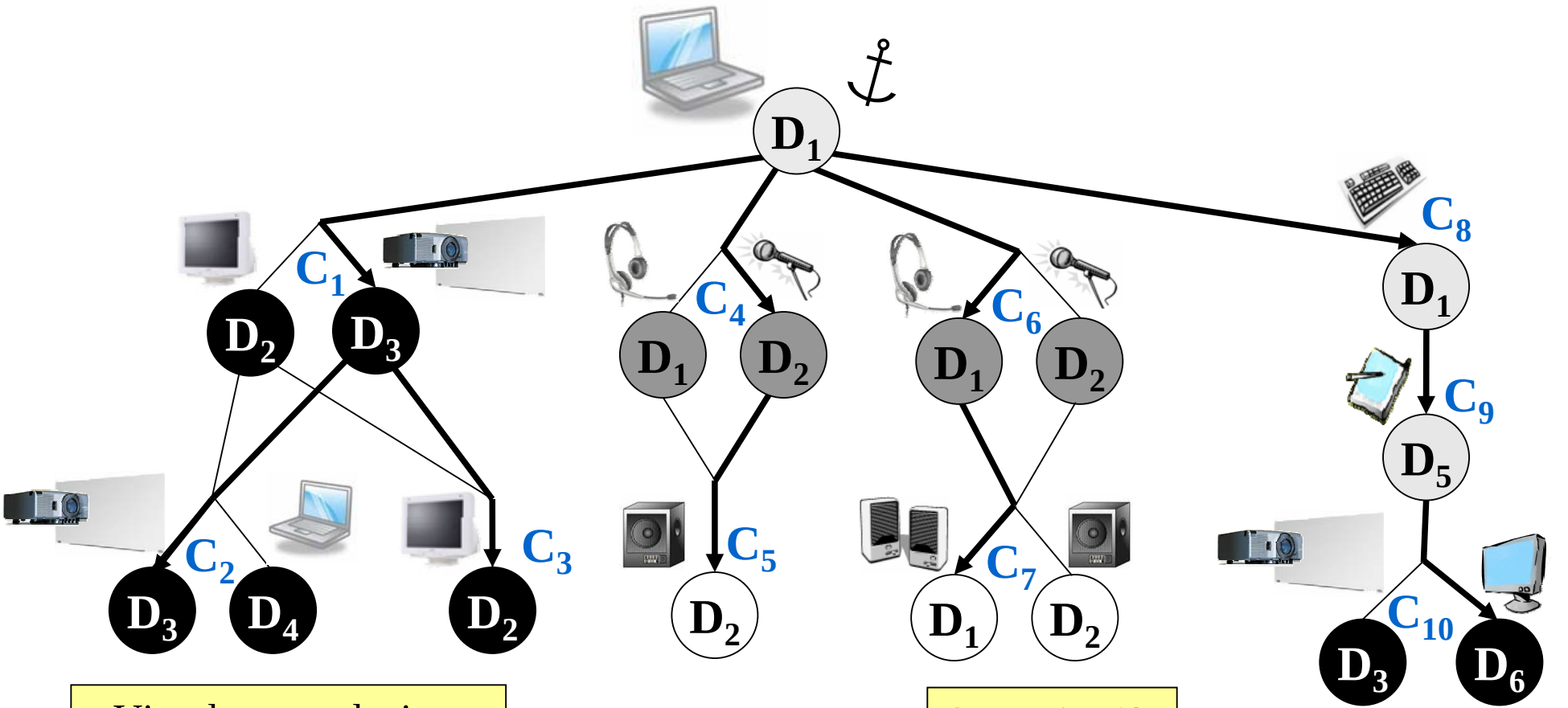


System Model

- Distributed Pervasive Application
  - Application modeled as a tree
    - Recursive dependencies
  - Containers
    - Represent devices
    - Unique ID
  - Components
    - Represent functionalities and resources
  - Directed contracts
    - Model interdependencies between components
    - Specify functionalities and properties of resources (e.g., display resolution)
    - May be multi-optional
  - All dependencies have to be resolved for a successful configuration



Exemplary Presentation Application



Visual output devices for presentation (slides, video, web conference)

Support for web conferences

Acoustic I/O devices for lecturer

Support of notes and drawings

Related Work

- Automatic configuration and adaptation of tree-based distributed applications
  - can be mapped to Constraint Satisfaction Problem (CSP)
- Algorithms from Artificial Intelligence suited for solving CSPs:
  - Arc Consistency algorithms ⇒ incomplete; inefficient or difficult to handle
  - min-conflicts algorithm ⇒ incomplete
  - Backtracking algorithms ⇒ complete
    - Overview of backtracking algorithms given by A. Baker
- Problems in existent backtracking algorithms:
  - Increased latency due to unnecessary adaptations (thrashing effect)
  - Exponential memory overhead due to stack that pops up during adaptation
  - Change of contract order to resolve conflicts

⇒ Direct Backtracking (DBT)

Direct Backtracking
General view

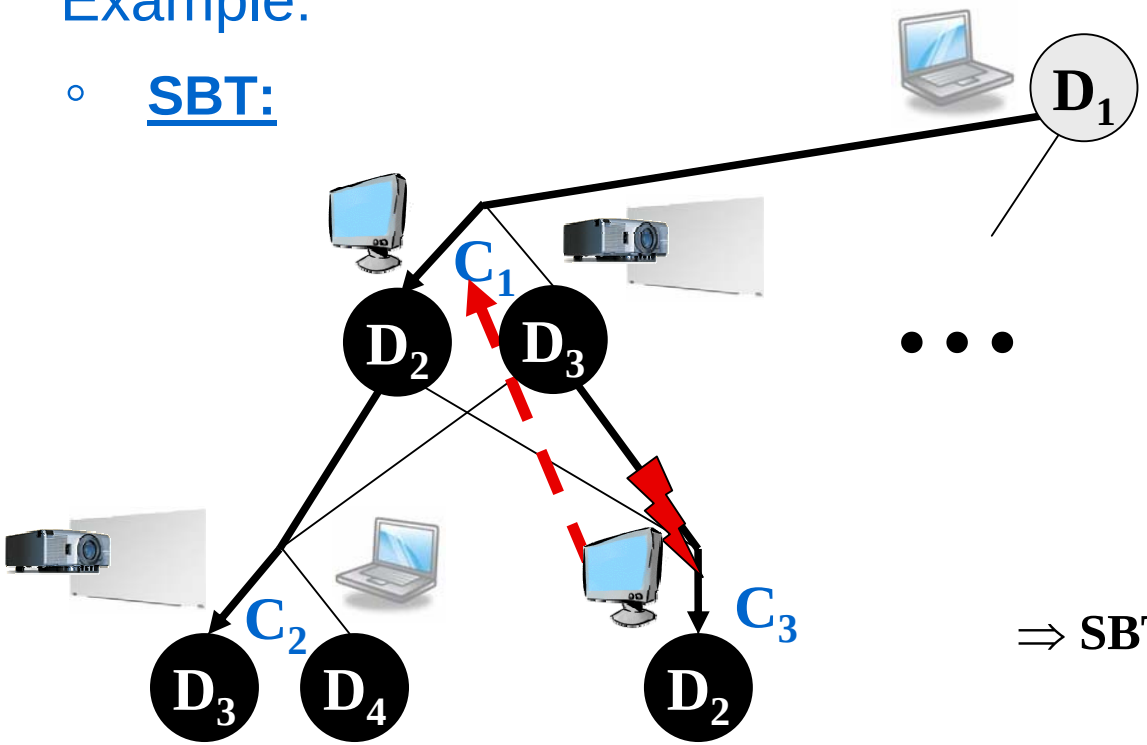
- Centralized configuration on devices with high computation power
- Resource information & distribution of configuration results
  - Specific protocols, independent from configuration process
  - Communication complexity linear in number of devices
- Same basic proceeding as Synchronous Backtracking (SBT)
  - Depth-first search in the tree of dependencies
  - Each component recurvisely instiantiates all of its child components
- SBT suffers from thrashing in adaptation processes

$\Rightarrow$ Two mechanisms in DBT to render adaptation more efficiently:

  - Proactive Backtracking Avoidance
  - Intelligent Backtracking

1st Drawback of Synchronous Backtracking

- Synchronous Backtracking
  - Suffers from many adaptation processes during configuration
  - However: Some of these adaptations could be avoided
- Example:
  - **SBT:**



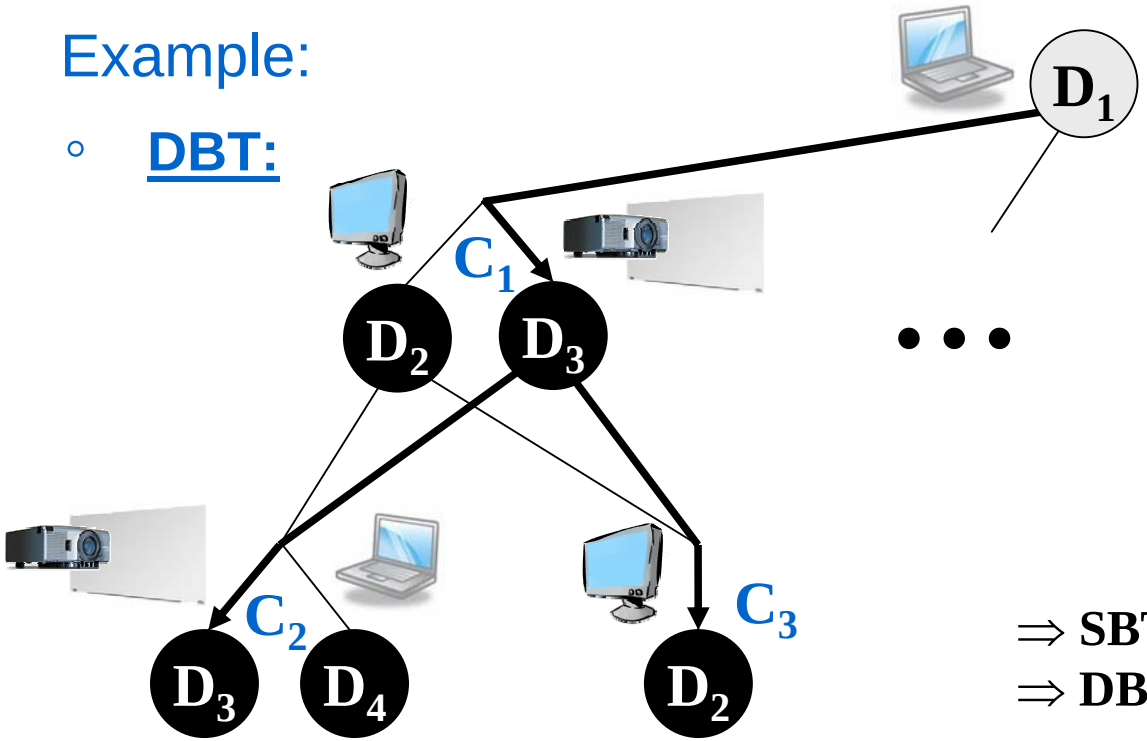
Available resources:

	D2	D3	D4
	0	-	-
	-	0	-
	-	-	1

⇒ **SBT: 2** adaptations

Direct Backtracking -
Proactive Backtracking Avoidance Mechanism

- Idea:
  - Carefully select option of a multi-optional contract to avoid backtracking
  - Keep flexibility for further configurations by instantiating components on devices with large amount of unused resources
- Example:
  - **DBT:**



Available resources:

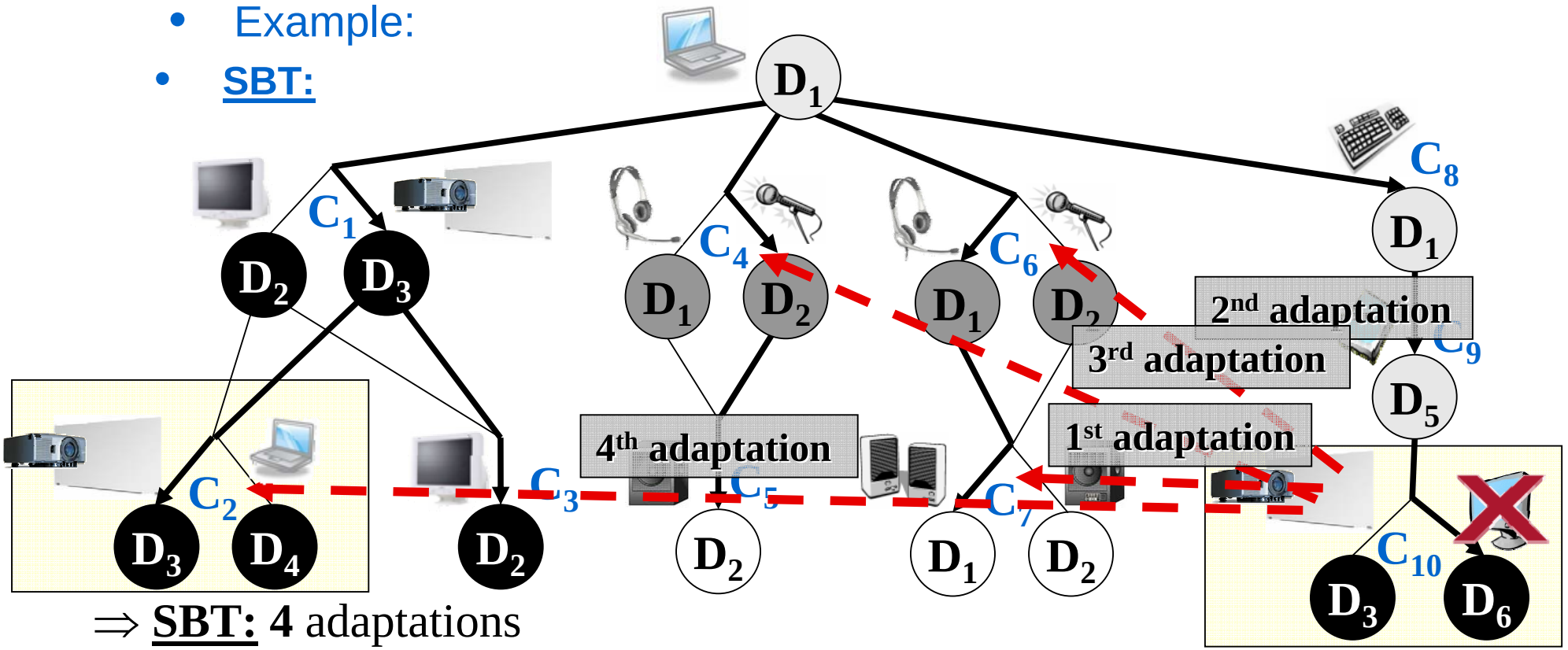
	D_2	D_3	D_4
	0	-	-
	-	**0**	-
	-	-	1

⇒ **SBT: 2** adaptations
⇒ **DBT: no** adaptation

2nd Drawback of Synchronous Backtracking

- Synchronous Backtracking
  - Does not perform adaptation processes in an intelligent way

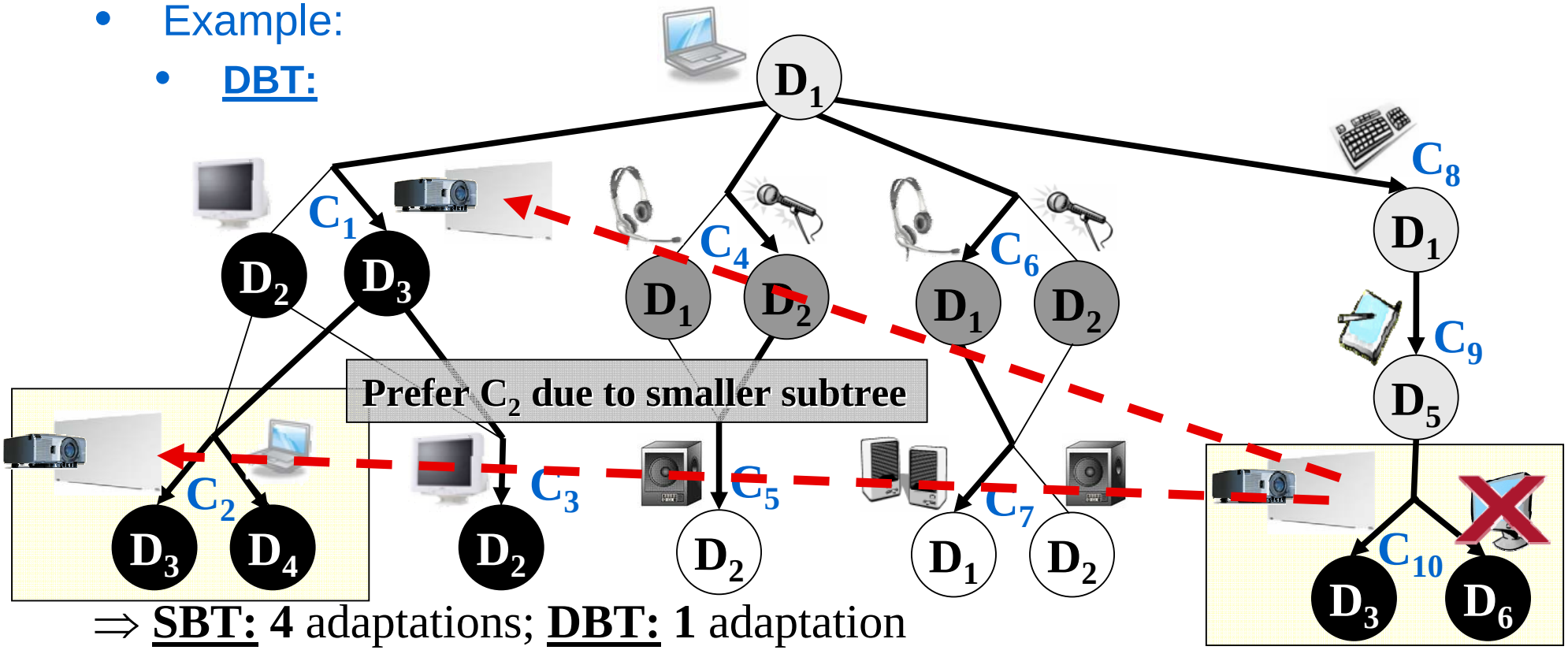
- Example:
- **SBT:**



⇒ **SBT:** 4 adaptations

Direct Backtracking -
Intelligent Backtracking Mechanism

- Idea:
  - Determine set of contracts that are suited for adaptation
  - Adapt contract with lowest overhead; keep intermediate results
- Example:
  - **DBT:**



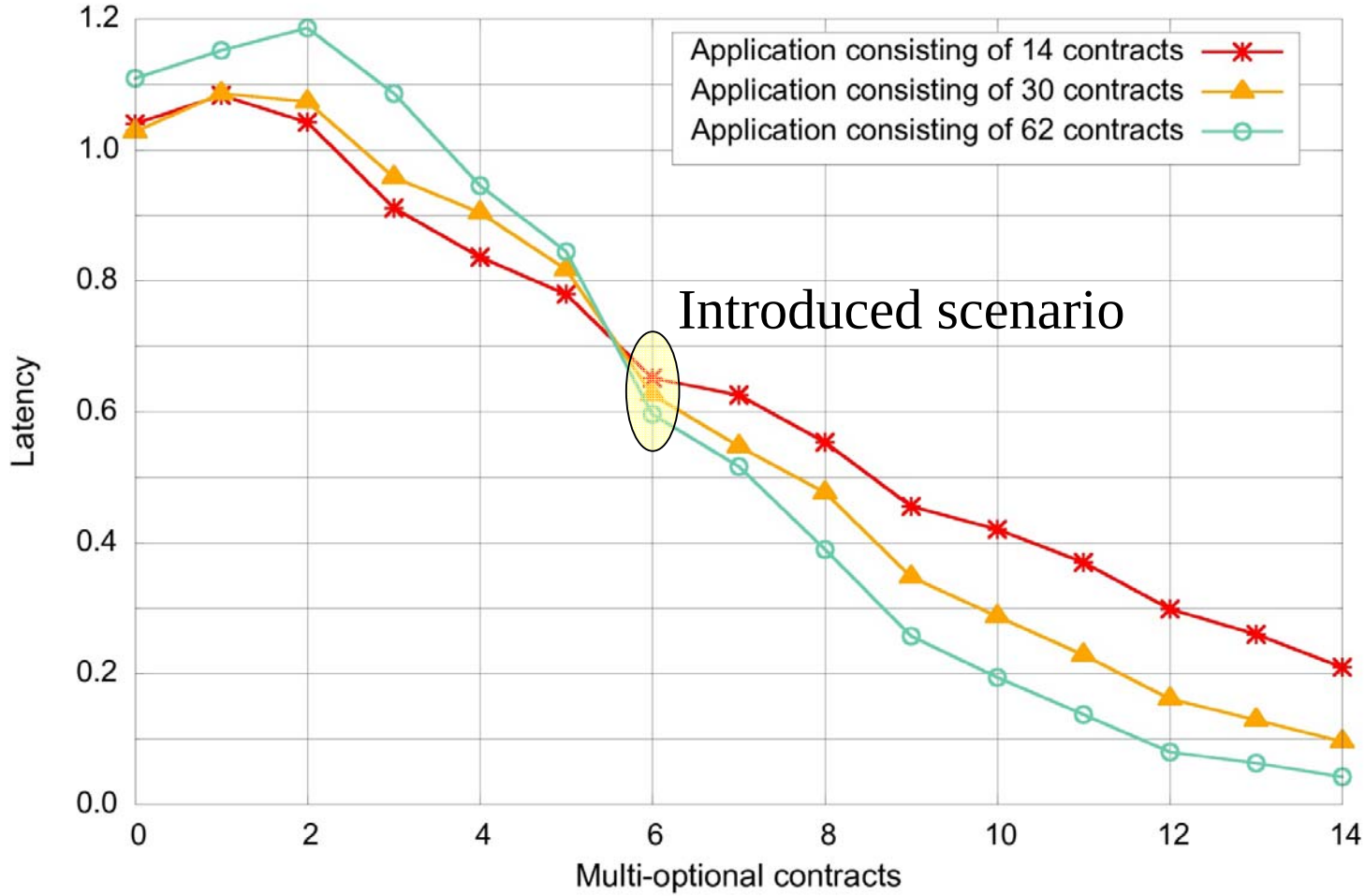
⇒ **SBT:** 4 adaptations; **DBT:** 1 adaptation

Evaluation Setup

- Discrete-event PCOM simulator
  - Becker et al.: PCOM - A Component System for Pervasive Computing. *International Conference on Pervasive Computing and Communications (PerCom) 2004,* Orlando, USA, 2004
- Evaluated applications:
  - Binary tree of height $2 \leq h \leq 10$, leading to $n = 2^{h+1} - 2$ components
  - Components placed in a round robin manner on $m \leq n$ containers
  - $k < n$ arbitrary components duplicated, placed on random containers
  - Containers got enough resources that at least one valid configuration exists
- 10,000 runs at different scenarios with DBT and SBT
  - Common desktop PC (Dual core CPU with 2.2 GHz, 2.0 GB RAM)

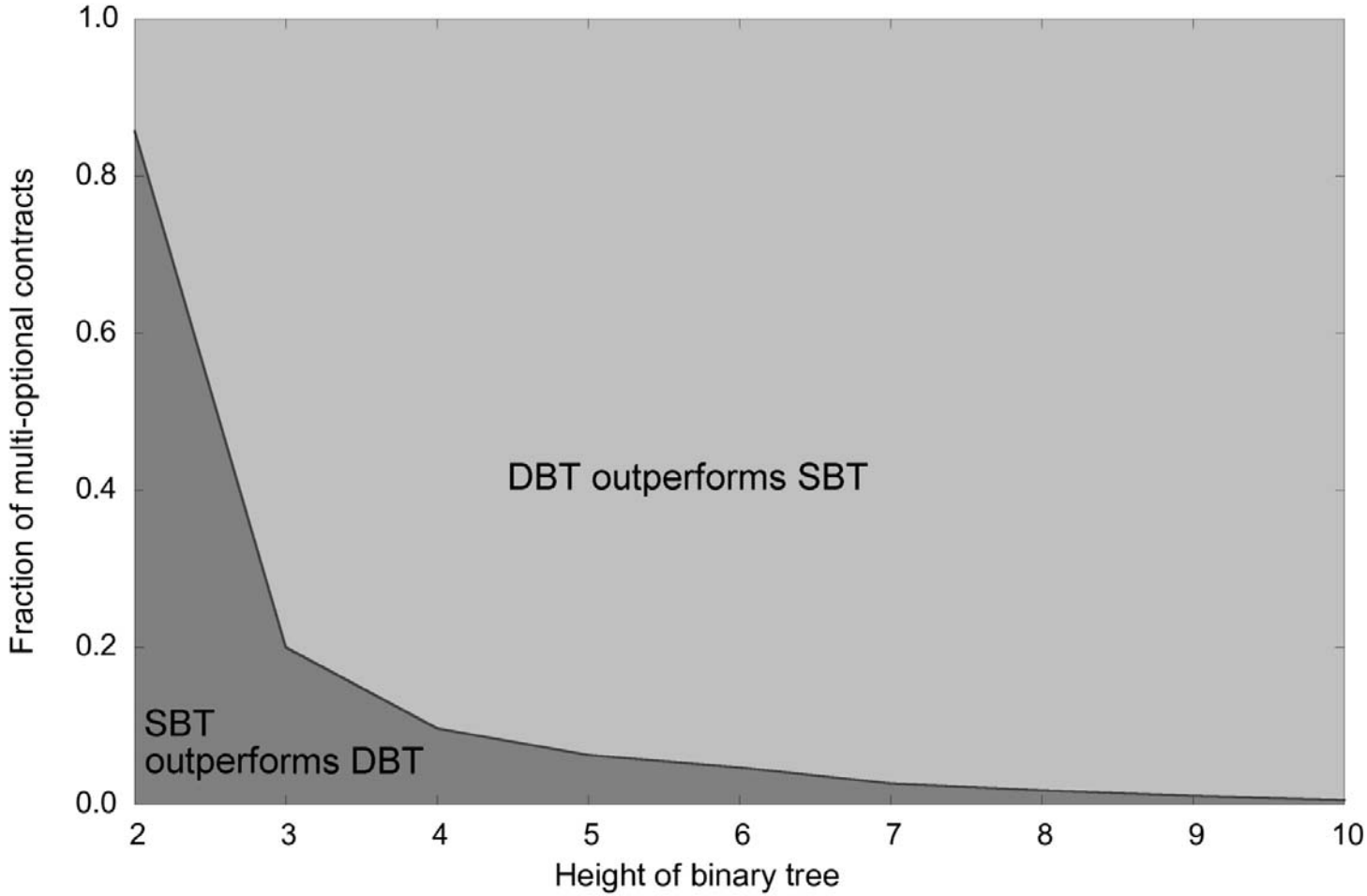
Evaluation Results (1)

- Configuration latency, relative to Synchronous Backtracking



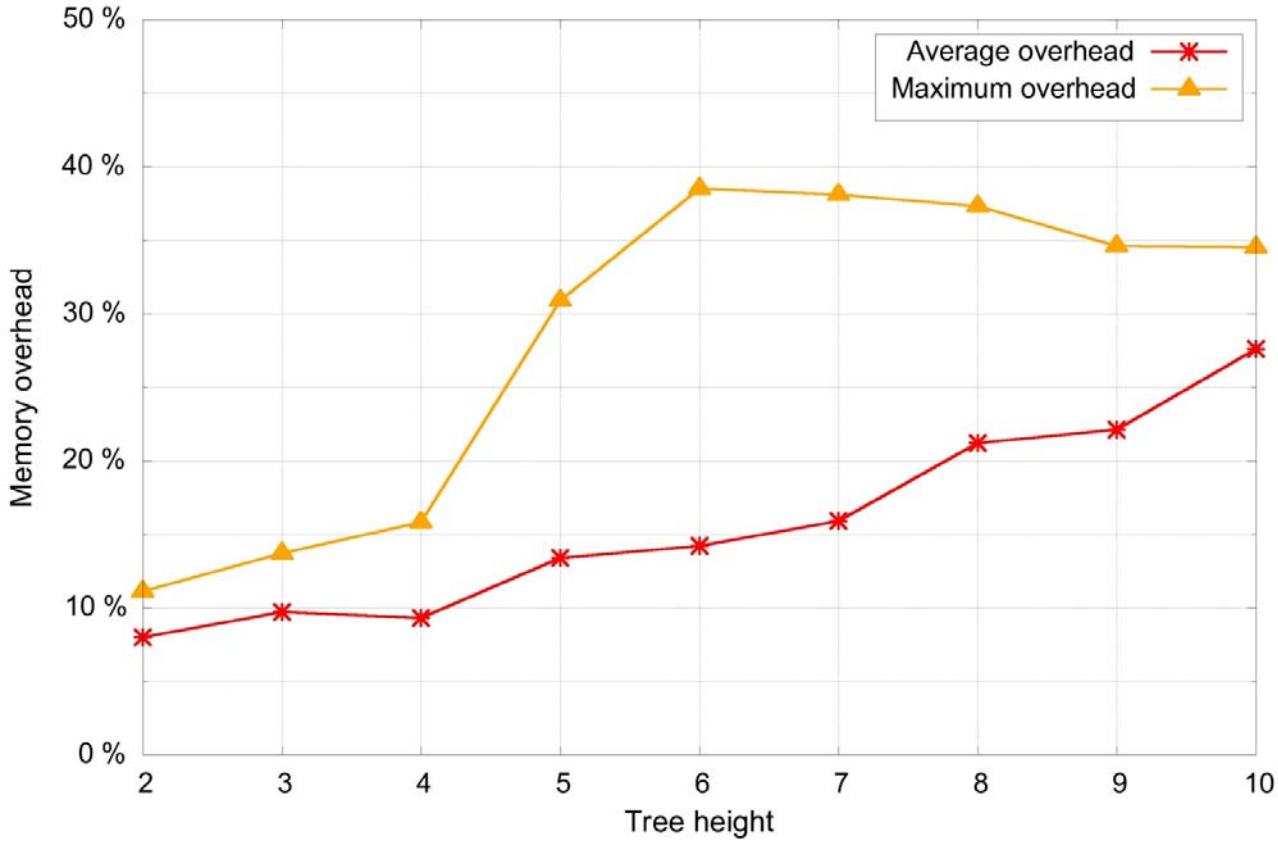
Evaluation Results (2)

- Break-even points [relative to application size and multi-optional contracts]



Evaluation Results (3)

- Memory overhead (RAM) [compared to Synchronous Backtracking]
  - Standard deviation: < 2 %



Conclusion

- Configuration and adaptation of Pervasive Applications highly relevant
- Direct Backtracking, an advanced algorithm for Pervasive Applications:
  - Proactive Backtracking Avoidance
  - Intelligent Backtracking
- Evaluation results
  - Configuration latencies
    - Significant reduction in most scenarios
    - Very low break-even points in case of huge applications
  - Memory overhead
    - Linear in the tree height (exponential in stack-based algorithms)
  - Communication complexity
    - Linear in the number of involved devices (square in distributed algorithms)

→ *Efficient and resource-conserving adaptation of Pervasive Applications!*

Questions?

Thank you for your attention!

Questions?

Contact:

Stephan Schuhmann
Institute of Parallel and Distributed Systems (IPVS)
Universität Stuttgart
Universitätsstraße 38
70569 Stuttgart, Germany

Email: stephan.schuhmann@ipvs.uni-stuttgart.de
Web: http://www.ipvs.uni-stuttgart.de/
Phone: (+49) 711 – 7816 442

References

- Arc Consistency algorithms
  - A. K. Mackworth: *Consistency in networks of relations.* Artificial Intelligence, vol. 8, pp. 99-118, 1977.
- min-conflicts algorithm
  - S. Minton, M. D. Johnston, A. B. Philips and P. Laird: *Minimizing Conflicts: A Heuristic Repair Method for Constraint Satisfaction and Scheduling Problems.* Artificial Intelligence, vol. 58, pp. 161-205, 1992.
- Backtracking algorithms
  - A. B. Baker: *Intelligent Backtracking on Constraint Satisfaction Problems: Experimental and Theoretical Results*. PhD Thesis, University of Oregon, 1995.