Utilizing Quad-Trees for Efficient Design Space Exploration with Partial Assignment Evaluation

Kai Neubauer[1], Philipp Wanko[2], Torsten Schaub[2], and Christian Haubelt[1]

[1]University of Rostock, Germany

[2]University of Potsdam, Germany

Universität Rostock

Traditio et Innovatio

Universität Potsdam

Introduction

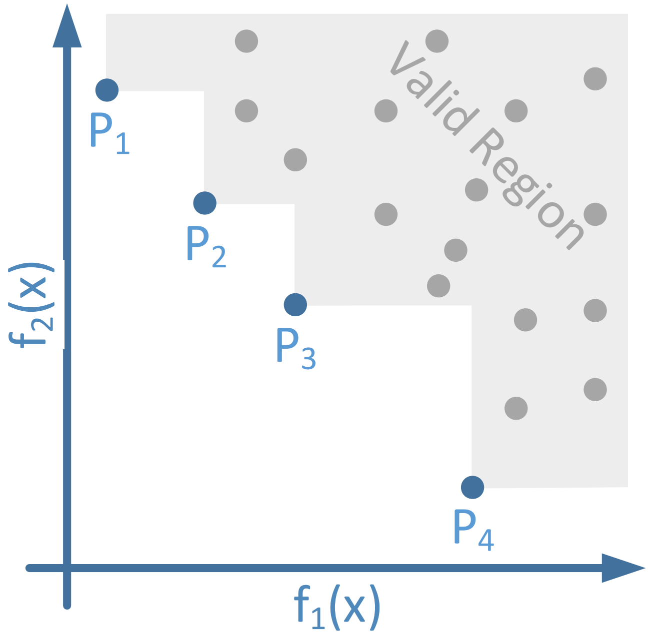
- System-level design is becoming more complex
  - Mapping, allocation, and scheduling
  - Heterogeneous processing platforms
- Aim is to find a system implementation that is…
  - valid regarding various constraints
  - optimal regarding quality properties
- Multiple approaches, e.g.:
  - Meta-Heuristics (MOEA, MOPSO, Ant colony, etc.)
  - Formal methods (SAT, ILP, ASP, etc.)
  - Hybrid techniques

Research Issues

- Meta-Heuristics not executed systematically
  - Combining previously found solutions
  - Tend to run into saturation
- Formal symbolic techniques
  - Find all solutions
  - Huge search space
- Problem: Enormous amount of comparisons
  - Fitness vectors stored in an archive
  - Dominance checks for novel solutions
  - Partial Assignment Evaluation

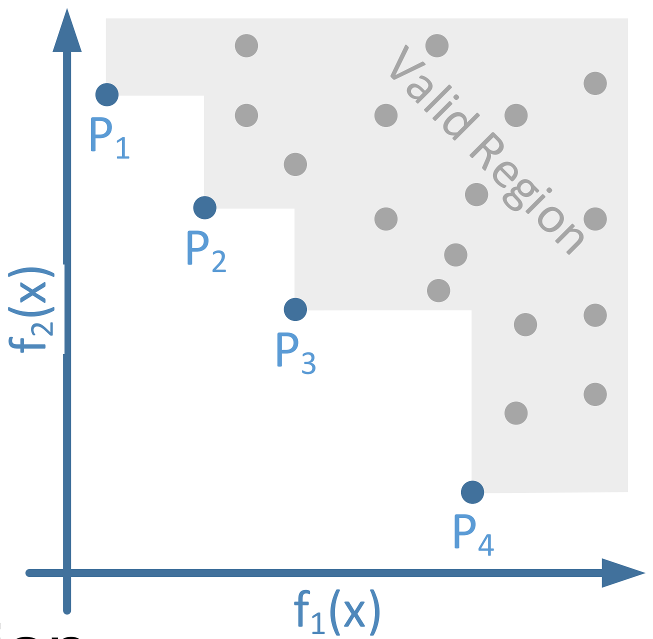
Multi-objective Optimization

- Design space exploration in order to find optimal solutions
- Optimality dependent on quality parameters
  - Often conflicting objectives – Multi-objective optimization Problem (MOOP)
  - Different design alternatives not totally ordered
  - Pareto optimality



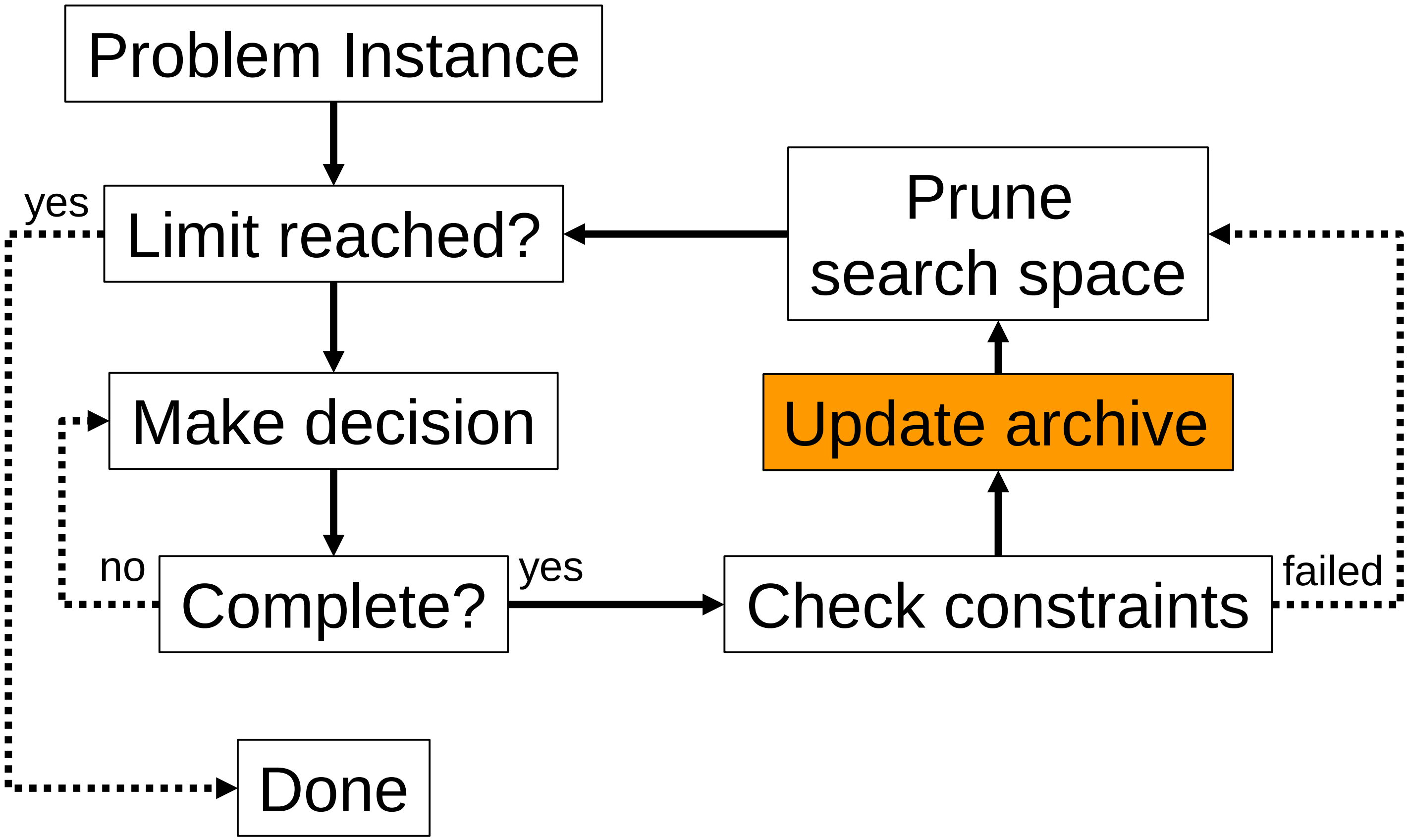
Multi-objective Optimization

- Dominance relations between $X = [x_i]$ and $Y = [y_i]$
  - X dominates Y
    - $\forall i: x_i \succcurlyeq y_i \wedge \exists i: x_i \succ y_i$
  - X is dominated by Y
    - $\forall i: x_i \preccurlyeq y_i \wedge \exists i: x_i \prec y_i$
  - X is incomparable to Y
    - $\exists i: x_i \succ y_i \wedge \exists i: x_i \prec y_i$



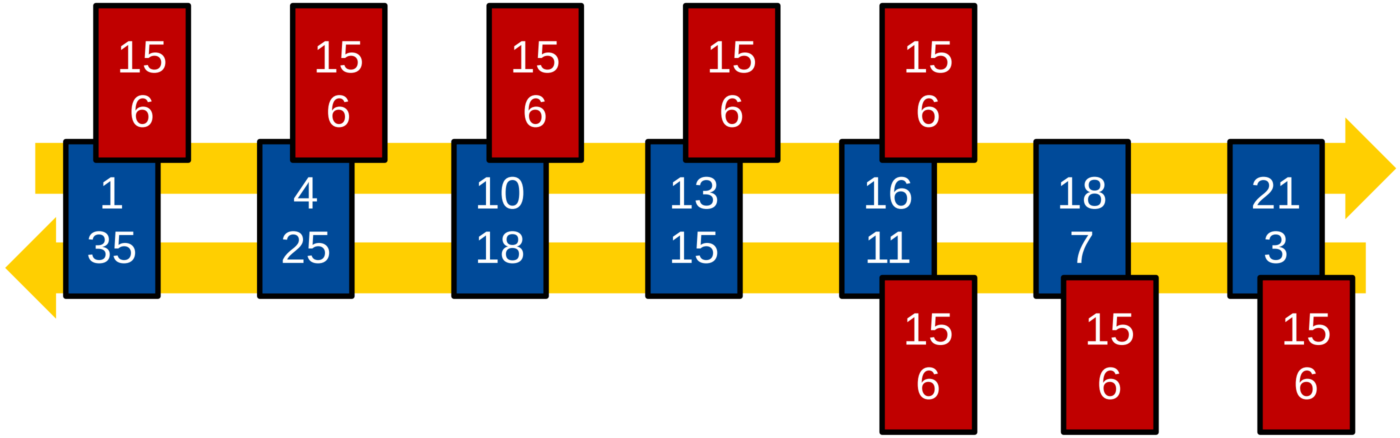
- X is Pareto optimal iff it is not dominated by any other solution
- Minimization is assumed in the following

Acquiring Pareto-Front



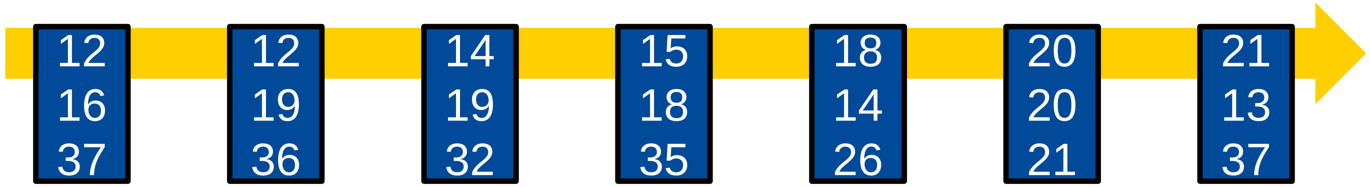
List-based Management

- Non-dominated solutions are saved to archive
- List-based approaches
  - O(n) for 2 dimensions – sorted for both dimensions



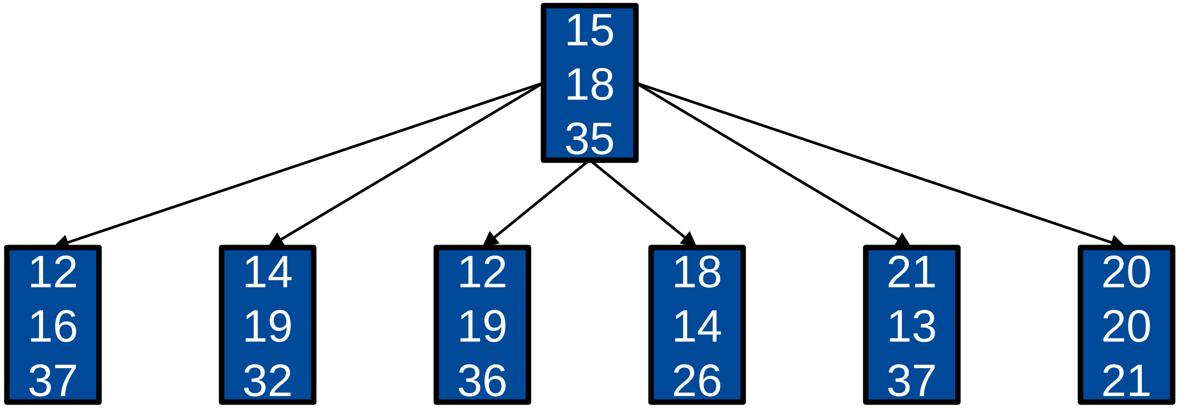
List-based Management

- Non-dominated solutions are saved to archive
- List-based approaches
  - O(n) for 2 dimensions – sorted for both dimensions
  - O(n*m) for more dimensions – only first dim. sorted



Tree-based Management

- Tree-based approaches
  - Each node represents one solution
  - Each solution is root to further solutions
  - “Ordered” by some degree
  - Comparing all solutions is unnecessary

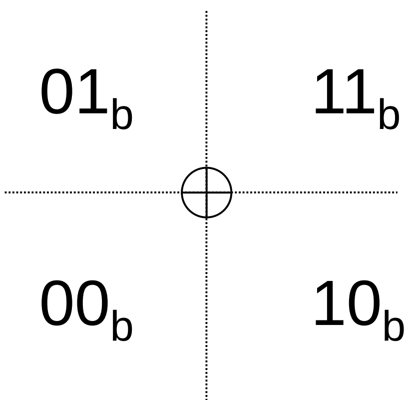


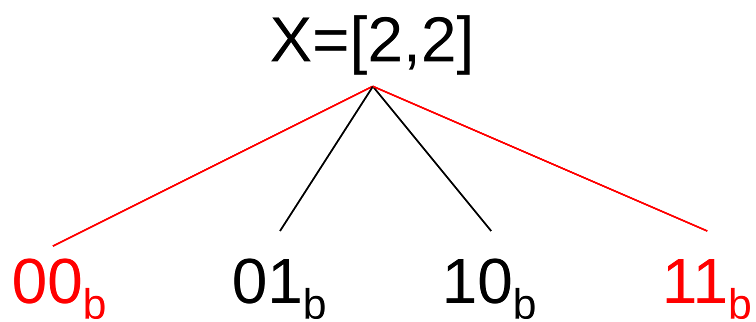
Quad-Trees

- Quad-Trees (QT) use b-tree structure
  - Each node represents one non-dominated solution
  - With m objectives : 2^m children
  - Children “0” and “m-1” are not saved
    - 0 dominates root
    - m-1 is dominated by root
- Each node is represented by its fitness vector

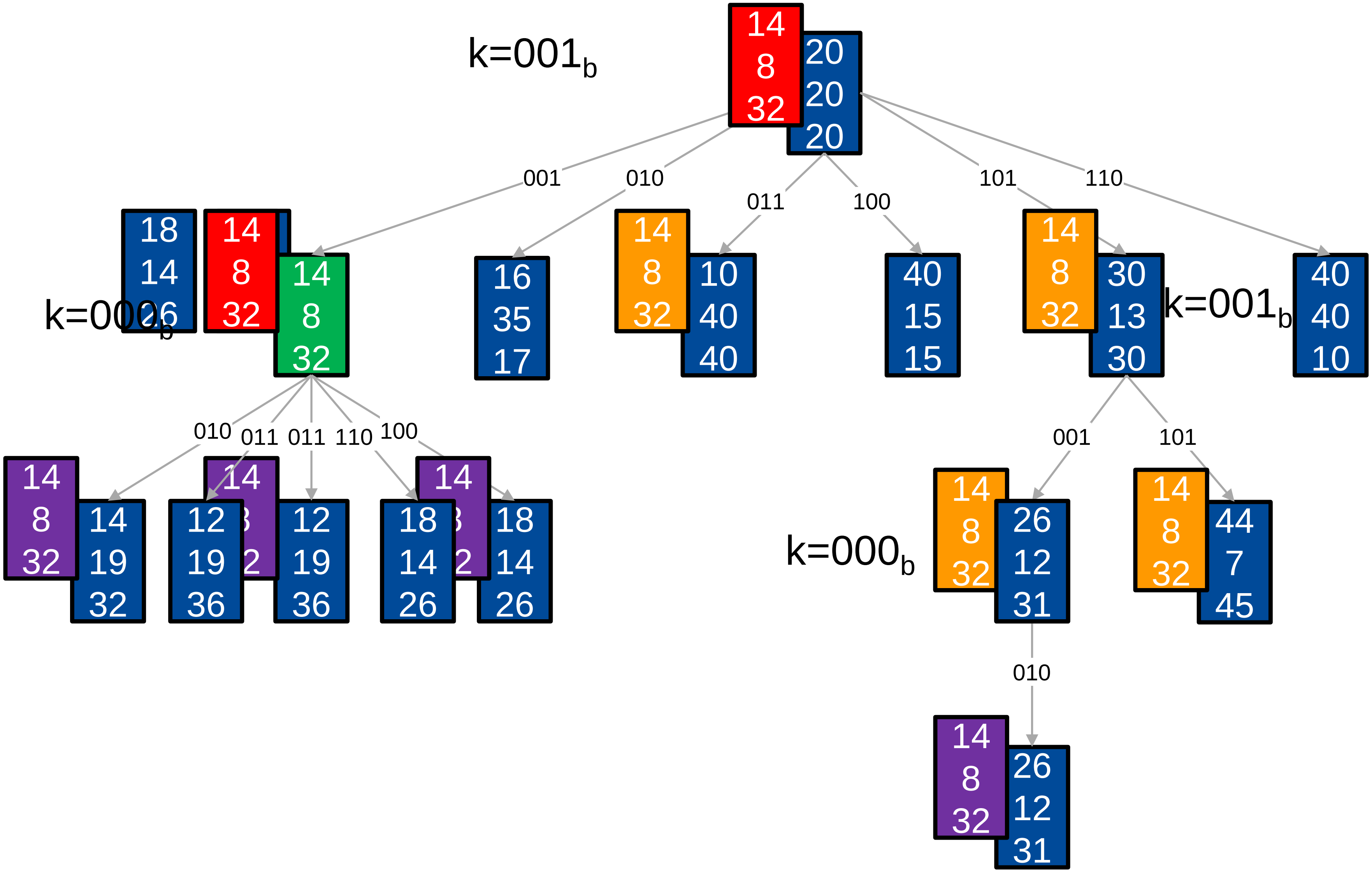
Quad-Trees

- New solutions may be added in one of the 2^m sub-trees
  - *k-successor* (m bit) determines the position a children is inserted
  - States which objective is better (0) or worse (1)
  - Position in m-dimensional coordinate system





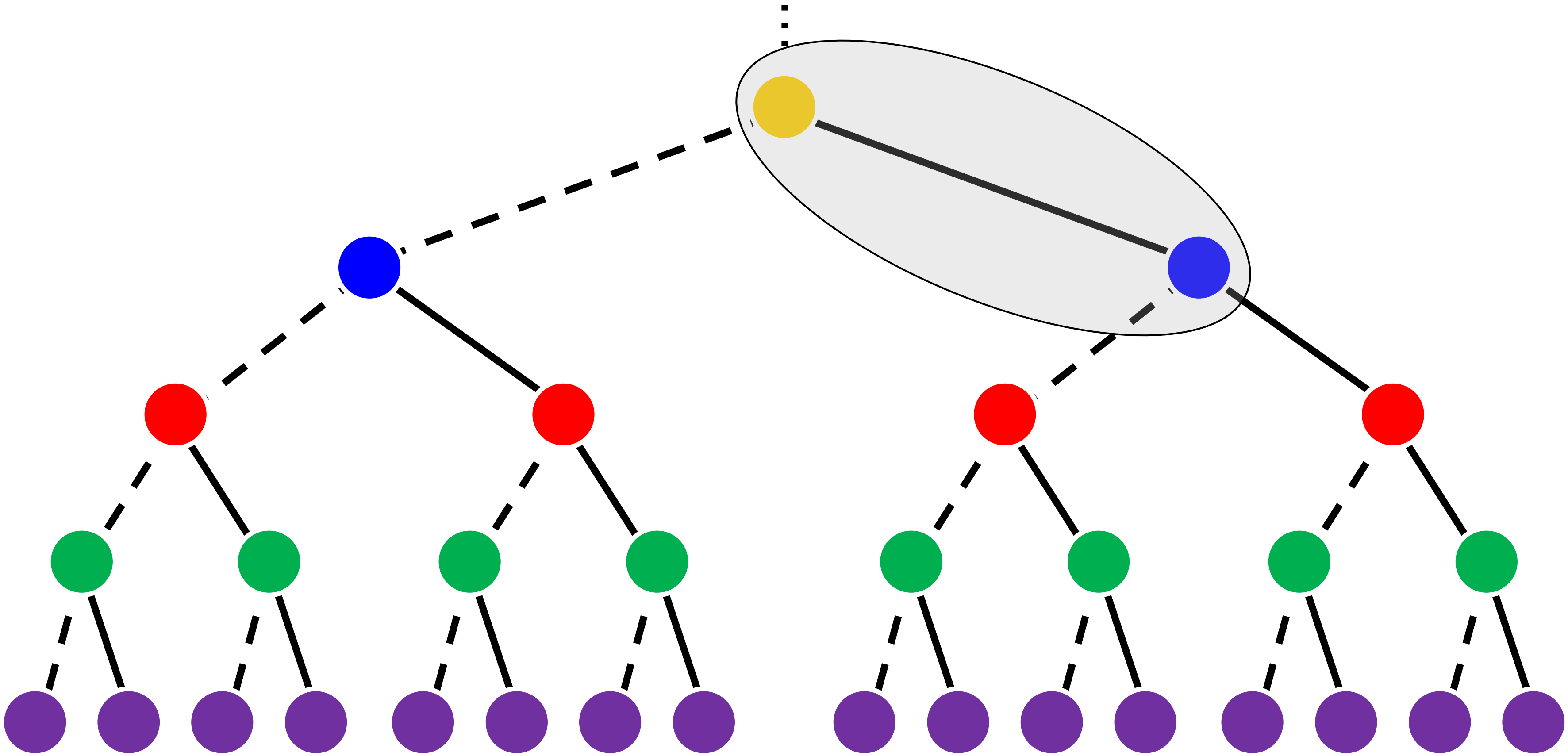
Updating Quad-Trees



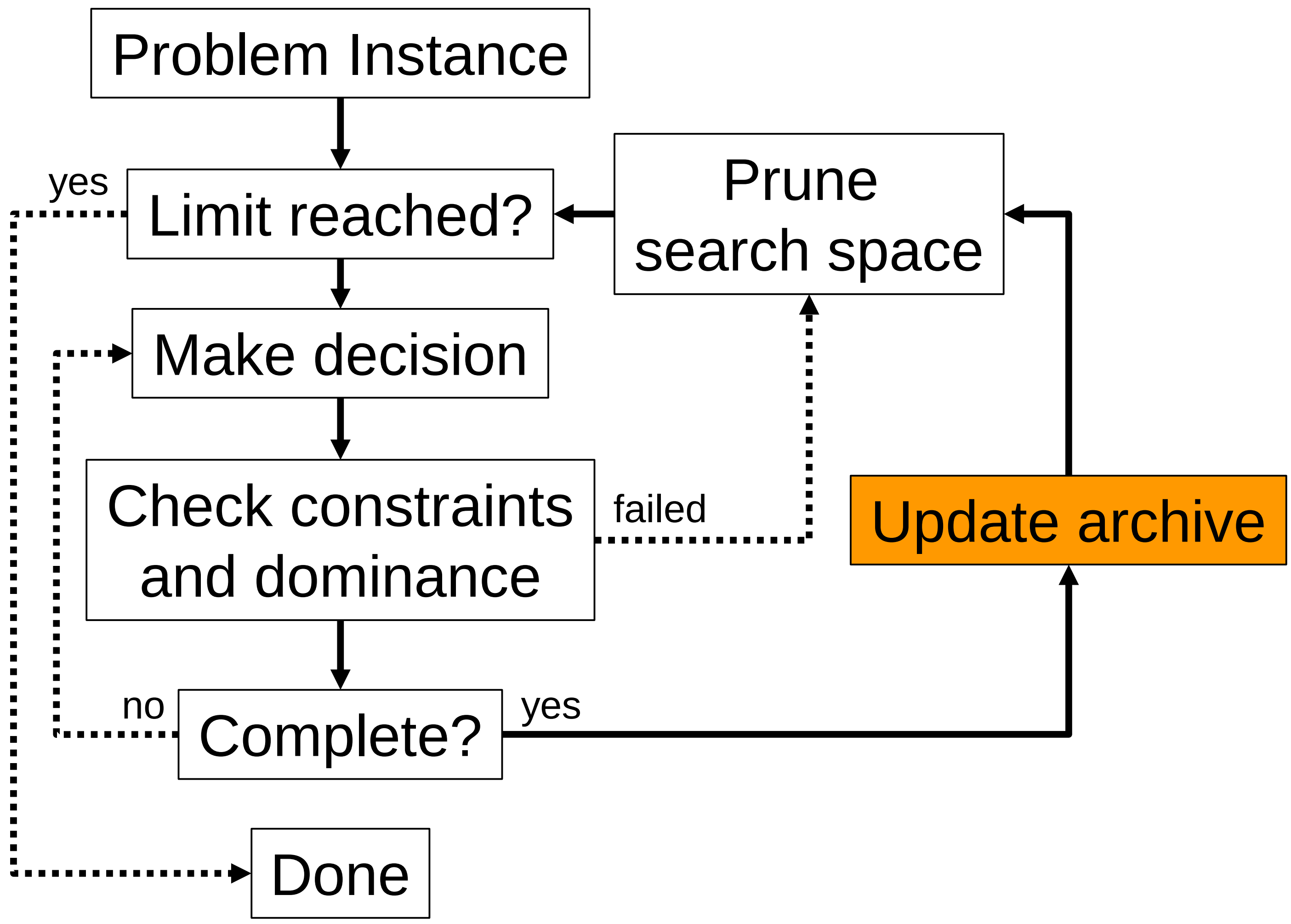
Comparison: List vs QT

List	Quad-Tree
▪ Simple implementation	▪ Constant complexity
▪ Good 2-D performance	▪ Fast dominance test
▪ Removing is easy	▪ Geometrically ordered
▪ Bad 3-D+ complexity	▪ Complex implementation
	▪ Removing is hard

Partial Assignment Evaluation



QTs For Partial Assignments

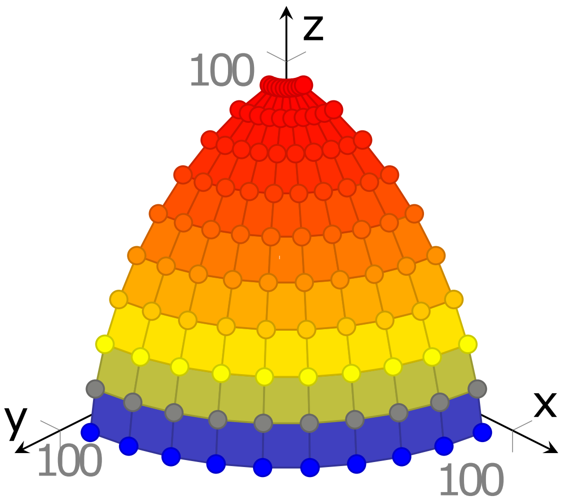


QTs For Partial Assignments

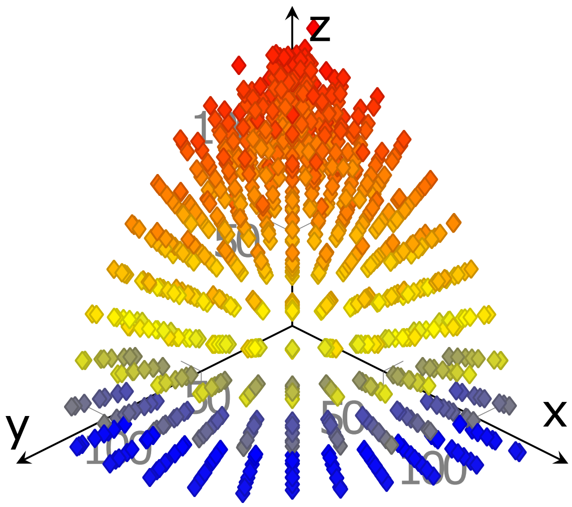
- Checking partial solutions is expensive
  - Each decision (set of decisions) has to be checked
- No need to check if partial solution dominates any other
  - Check if it is already dominated
  - Archive is only updated for complete solutions
- Expensive operations are only executed once
- Ratio Checking to Updating increases

Evaluation

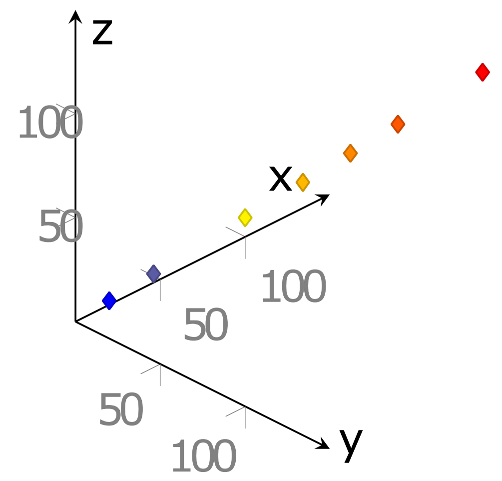
- Setup:
  - Implementation in Python 2.7
  - Spherical Pareto-Front (2 to 5 Dimensional) – Fig. (a)
  - Dominated solutions – Fig. (b)
  - 50 to 200 partial solutions – Fig. (c)



(a)



(b)



(c)

Evaluation (cont.)

Evaluation (cont.)

Conclusion

- Quad-Trees for formal methods with Partial Assignment Evaluation
- Dominance Checks performed for each partial solution
- Quad-Trees offer a fast dominance identification
- Significantly lower number of comparisons

- Future Work:
  - Balancing algorithms
  - Use structural information for steering solving process

References

1. W. Habenicht. Essays and Surveys on Multiple Criteria Decision Making, chapter *Quad Trees, a Datastructure for Discrete Vector Optimization Problems*, pages 136–145. Springer Berlin Heidelberg, 1983.
2. S. Mostaghim and J. Teich. Evolutionary Multiobjective Optimization, chapter *Quad-trees: A Data Structure for Storing Pareto Sets in Multiobjective Evolutionary Algorithms with Elitism*, pages 81–104. Springer London, 2005.