

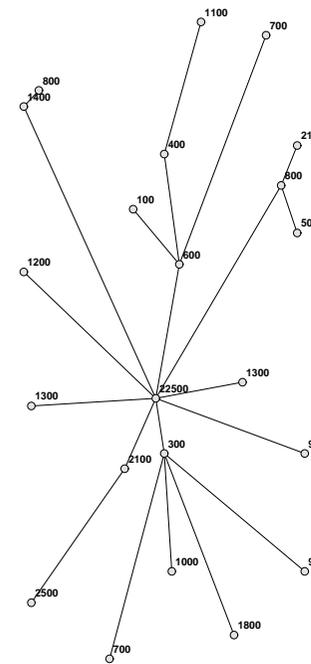
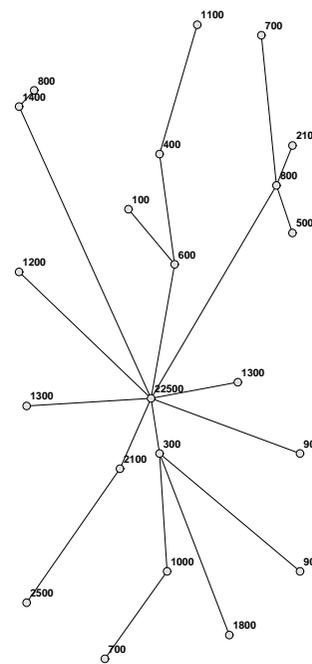
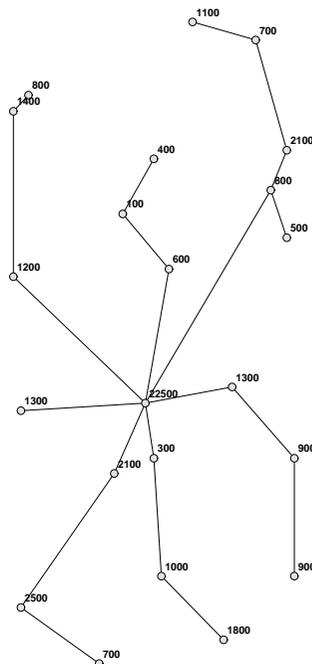
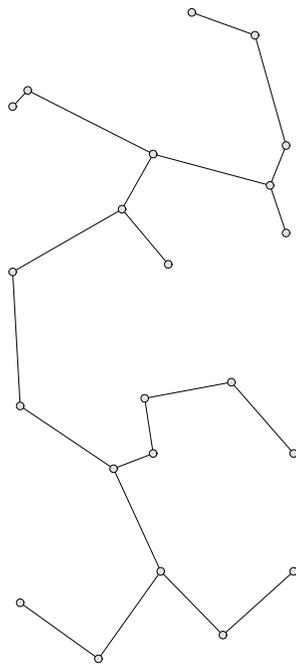
Using Cyberinfrastructure Tools to Solve Biobjective Integer Programs

Ted Ralphs, Menal Guzelsoy, and Jeff Linderoth

COR@L Lab, Industrial and Systems Engineering, Lehigh University

Matthew Saltzman and Margaret Wiecek

Mathematical Sciences, Clemson University



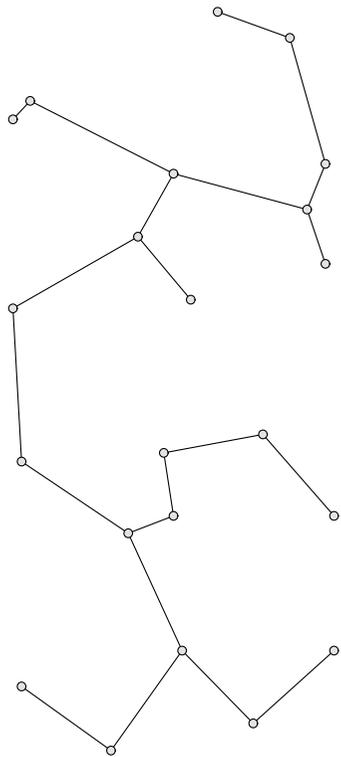
Outline of Talk

- Motivation
- Preliminaries
- The WCN Algorithm
- Implementation
- Example
- Computational Results

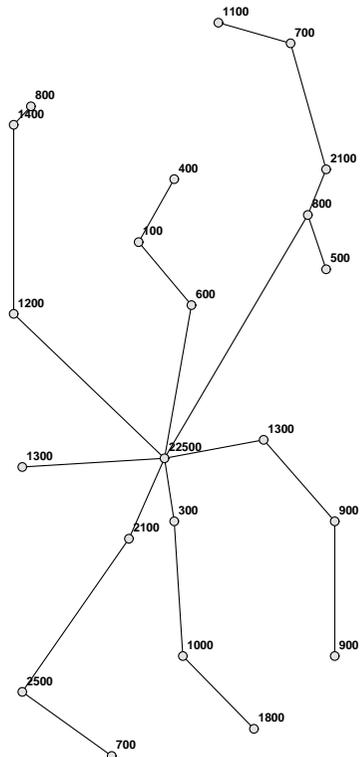
Motivation: Cable-Trench Problems

- A single commodity must be supplied to a set of customers from a central supply point.
- We want to design a network, possibly obeying capacity and other side constraints.
- In the **Cable-Trench Problem**, we consider both
 - the **cost of construction** (the sum of lengths of all links), and
 - the **latency of the resulting network** (the sum of length multiplied by demand carried for all links).
- These are competing objectives for which we would like to analyze the **tradeoff**.
- We can formulate this problem as a **biobjective integer program**.

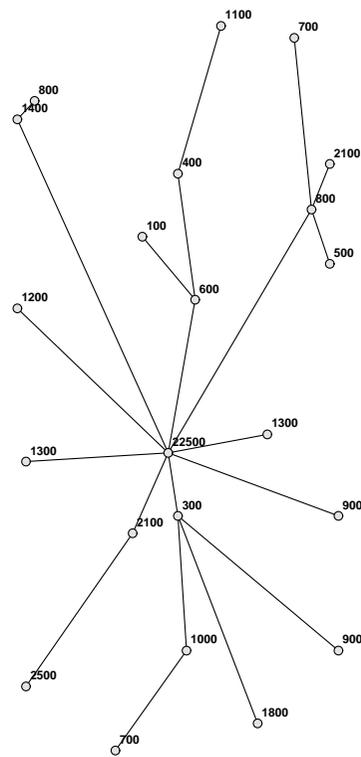
Solutions for a Small CTP Instance



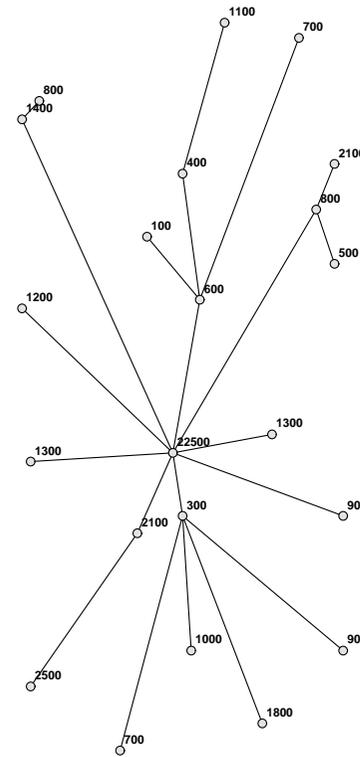
(a)



(b)



(c)



(d)

Biobjective Mixed-integer Programs

A *biobjective* or *bicriterion mixed-integer program* (BMIP) is an optimization problem of the form

$$\begin{array}{ll} \text{vmax} & f(x) \\ \text{subject to} & x \in X, \end{array}$$

where

- $f : \mathbb{R}^n \rightarrow \mathbb{R}^2$ is the *(bicriterion) objective function*, and
- $X \subset \mathbb{Z}^p \times \mathbb{R}^{n-p}$ is the *feasible region*, usually defined to be

$$\{x \in \mathbb{Z}^p \times \mathbb{R}^{n-p} \mid g_i(x) \leq 0, i = 1, \dots, m\}$$

for functions $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, \dots, m$.

The *vmax* operator indicates that we are interested in generating the *efficient solutions* (defined next).

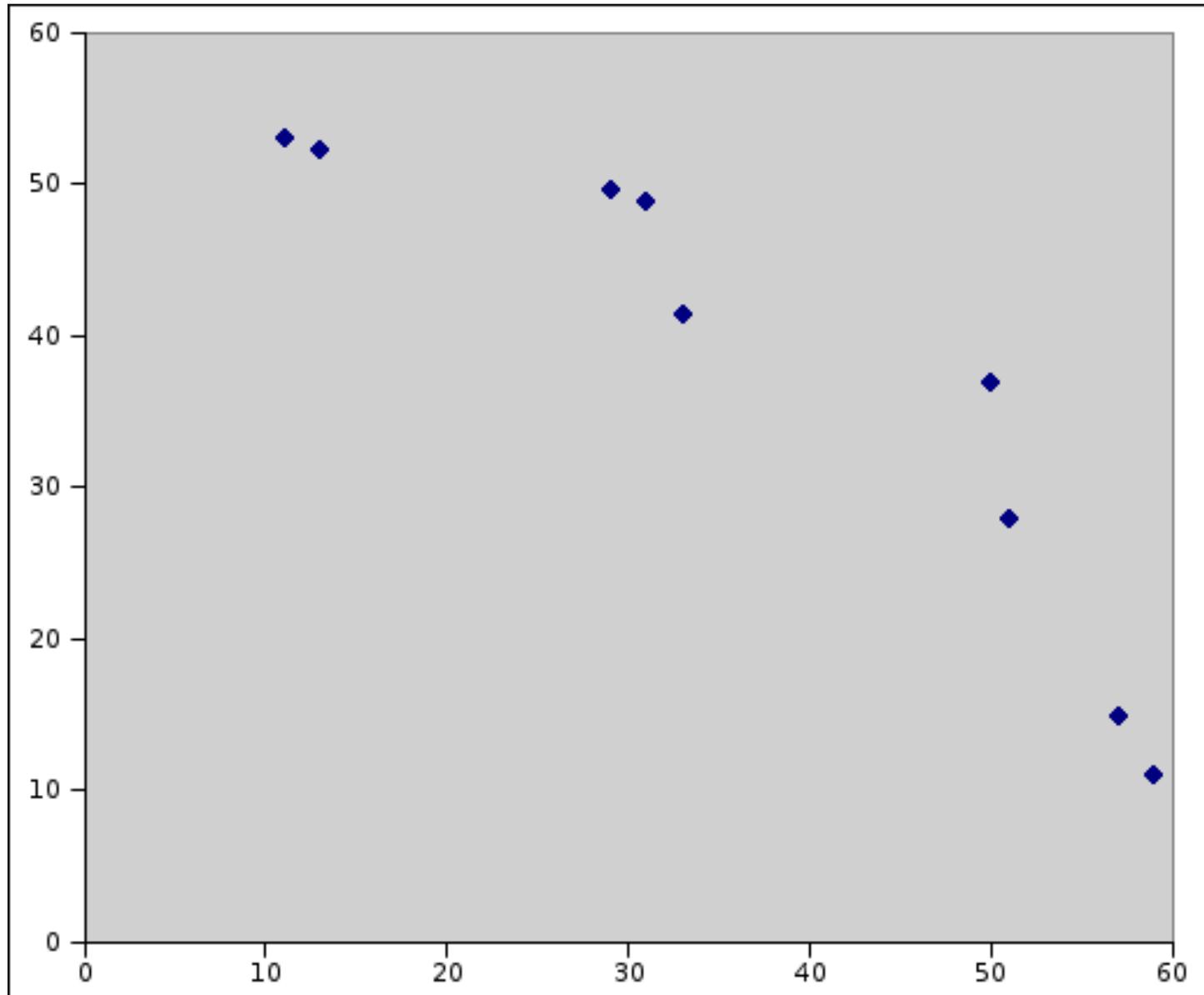
Some Definitions

- $x^1 \in X$ *dominates* $x^2 \in X$ if $f_i(x_1) \geq f_i(x_2)$ for $i = 1, 2$ and at least one inequality is strict.
- If both inequalities are strict, the dominance is *strong* (otherwise *weak*).
- Any $x \in X$ not dominated by any other member of X is said to be *efficient*.
- The set of *outcomes* is defined to be $Y = f(X) \subset \mathbb{R}^2$.
- In outcome space, BMIP can be restated as

$$\begin{array}{ll} \text{vmax} & y \\ \text{subject to} & y \in f(X), \end{array}$$

- If $x \in X$ is efficient, then $y = f(x)$ is *Pareto*.
- For simplicity, we work in outcome space.
- Our goal is to generate the set of **all Pareto outcomes**.

Illustrating Pareto Outcomes



Probing Algorithms

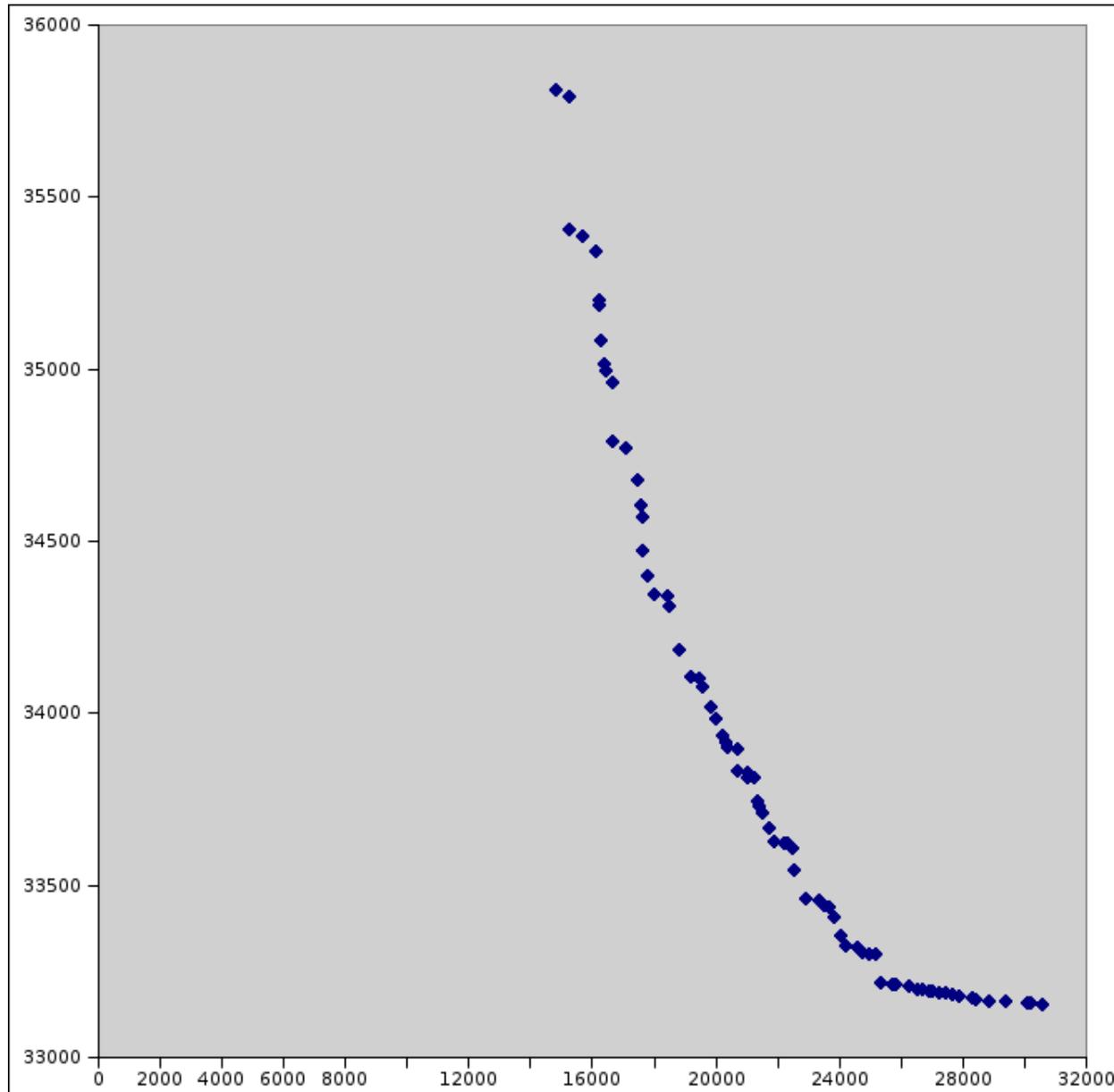
- A wide array of algorithms for generating Pareto outcomes have been proposed.
- We will focus on *probing algorithms* that *scalarize* the objective, i.e., replace it with a single criterion.
- Such algorithms reduce solution of a BMIP to a series of MIPs.
- The main factor in the running time is the number of *probes*.
- The most obvious scalarization is the *weighted sum objective*.
- We replace the original objective with

$$\max_{y \in f(X)} \beta y_1 + (1 - \beta) y_2$$

to obtain a parameterized family of MIPs.

- Such a subproblem can only generate *supported outcomes*.

Example: Pareto Outcomes for Cable-Trench Instance



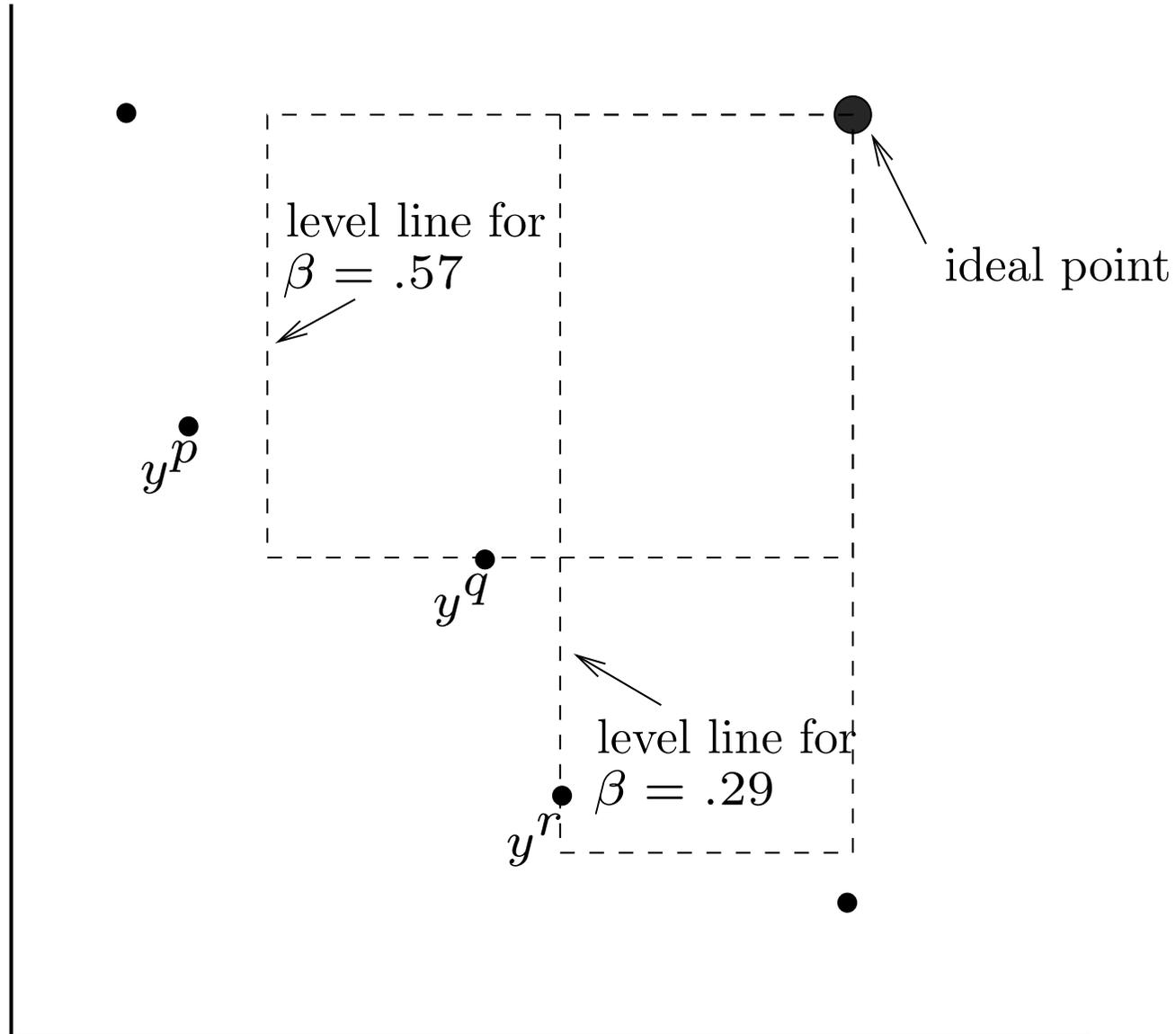
The Weighted Chebyshev Norm

- To generate unsupported outcomes, we replace the weighted sum objective with a *weighted Chebyshev norm* (WCN) objective.
- The *Chebyshev norm* (l_∞ norm) in \mathbb{R}^2 is defined by $\|y\|_\infty = \max\{|y_1|, |y_2|\}$.
- The *weighted Chebyshev norm* with weight $0 \leq \beta \leq 1$ is defined by $\|y\|_\infty = \max\{\beta|y_1|, (1 - \beta)|y_2|\}$.
- The *ideal point* y^* is (y_1^*, y_2^*) where $y_i^* = \max_{x \in X} (f(x))_i$.
- Methods based on the WCN select outcomes with minimum WCN distance from the ideal point by solving

$$P(\beta) = \min_{y \in f(X)} \{\|y^* - y\|_\infty^\beta\}. \quad (1)$$

- **Bowman** (1976) showed that every Pareto outcome is a solution to (1) for some $0 \leq \beta \leq 1$.
- The converse only holds if the instance is **uniformly dominant**.

Illustrating the WCN



Solving $P(\beta)$

- Problem (1) is equivalent to

$$\begin{array}{ll} \text{minimize} & z \\ \text{subject to} & z \geq \beta(y_1^* - y_1), \\ & z \geq (1 - \beta)(y_2^* - y_2), \text{ and} \\ & y \in f(X). \end{array} \quad (2)$$

- This is a MIP, which can be solved by standard methods.
- Note that this reformulation can produce weakly dominated outcomes.

Ordering the Pareto Outcomes

- **Eswaran** (1989) suggested ordering the Pareto outcomes so that
 - $Y_E = \{y_p \mid 1 \leq p \leq N\}$, and
 - if $p < q$, then $y_1^p < y_1^q$ (and hence $y_2^p > y_2^q$).
- For any Pareto outcome y_p , if we define

$$\beta_p = (y_2^* - y_2^p) / (y_1^* - y_1^p + y_2^* - y_2^p),$$

then y^p is the unique optimal outcome for (1) with $\beta = \beta_p$.

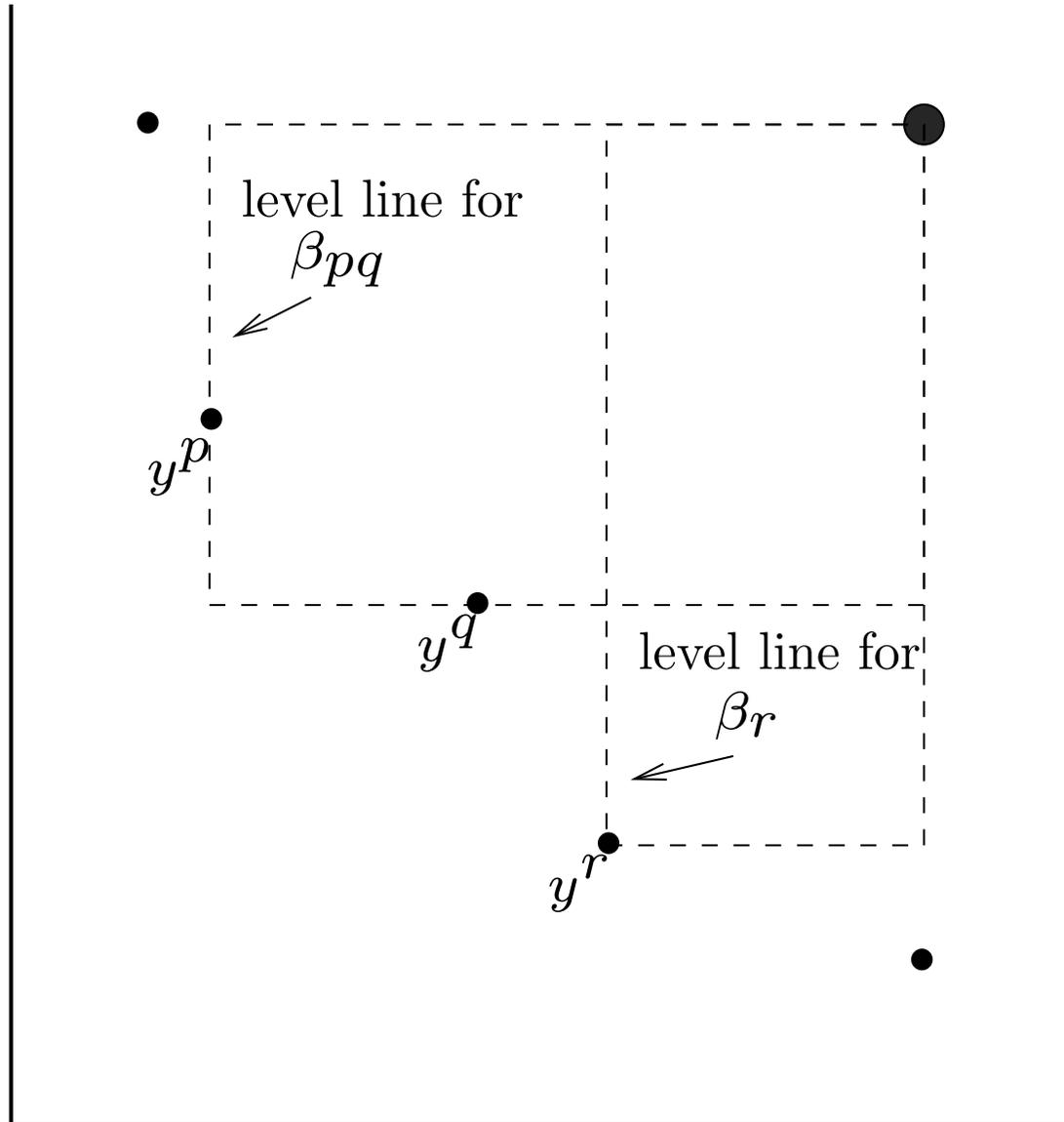
- For any pair of Pareto outcomes y^p and y^q with $p < q$, if we define

$$\beta_{pq} = (y_2^* - y_2^q) / (y_1^* - y_1^p + y_2^* - y_2^q), \quad (3)$$

then y^p and y^q are both optimal outcomes for (1) with $\beta = \beta_{pq}$.

- This provides us with a notion of *adjacency* and *breakpoints*.

Breakpoints Between Pareto Outcomes with the WCN



Algorithms Based on the WCN

- **Eswaran** (1989) proposed an algorithm based on binary search over the values of β , but the number of probes can be prohibitive.
- **Solanki** (1991) proposed an algorithm to generate an approximation to the Pareto set using the WCN.
- The **Solanki** algorithm probes between pairs of known outcomes using a procedure similar to that of **Chalmet**.
- We propose an algorithm that extends Solanki's ideas.
- The **WCN Algorithm**
 - is based on standard MILP solution techniques,
 - can produce all Pareto outcomes with $2N - 1$ probes, and
 - can produce the breakpoints between solutions.

The WCN Algorithm

Initialization Solve $P(1)$ and $P(0)$ to identify optimal outcomes y^1 and y^N , respectively, and the ideal point $y^* = (y_1^1, y_2^N)$. Set $I = \{(y^1, y^N)\}$.

Iteration While $I \neq \emptyset$ do:

1. Remove any (y^p, y^q) from I .
2. Compute β_{pq} as in (3) and solve $P(\beta_{pq})$. If the outcome is y^p or y^q , then y^p and y^q are adjacent in the list (y^1, y^2, \dots, y^N) .
3. Otherwise, a new outcome y^r is generated. Add (y^p, y^r) and (y^r, y^q) to I .

This reduces solution of the original BMIP to solution of a sequence of $2N - 1$ subproblems (under the assumption of uniform dominance).

Implementation

- A variety of algorithms for bicriteria optimization have been implemented as extensions to the **SYMPHONY** callable library.
- The subproblems are solved using a modified version of branch and cut.
 - The user specifies a second objective.
 - When using the WCN, **SYMPHONY** performs the required reformulation.
 - **SYMPHONY** employs one of several methods for eliminating weakly dominated solutions.
- Solver features
 - Can produce approximations to the Pareto set.
 - Implements several well-known algorithms for enumerating the Pareto set.
 - Can warm start subproblems.
 - Can maintain a global cut pool between iterations.
- Available from COIN-OR (projects.coin-or.org/SYMPHONY).

Parallelizing the WCN Algorithm

- Enumerating the entire Pareto set can be extremely difficult for hard combinatorial problems.
- The **WCN algorithm** is, however, naturally **parallelizable**.
- A simple **master-worker** implementation
 - The **master** keeps a **queue** of subproblems to be solved.
 - When a **worker** becomes free, the **master** picks a subproblem off the queue and sends it to the **worker**.
 - The **worker** returns either
 - * Message that the subproblem is **infeasible** (a new breakpoint).
 - * Two **new subproblems** to be added to the queue.
 - Continue until the queue is empty.
- This algorithm is a perfect candidate for solving on the **computational grid**.
 - It is coarse-grained and asynchronous.
 - Subproblem descriptions consist of only a few parameters.
 - Only the list of **breakpoints** and **solutions** generated so far are needed to restart.

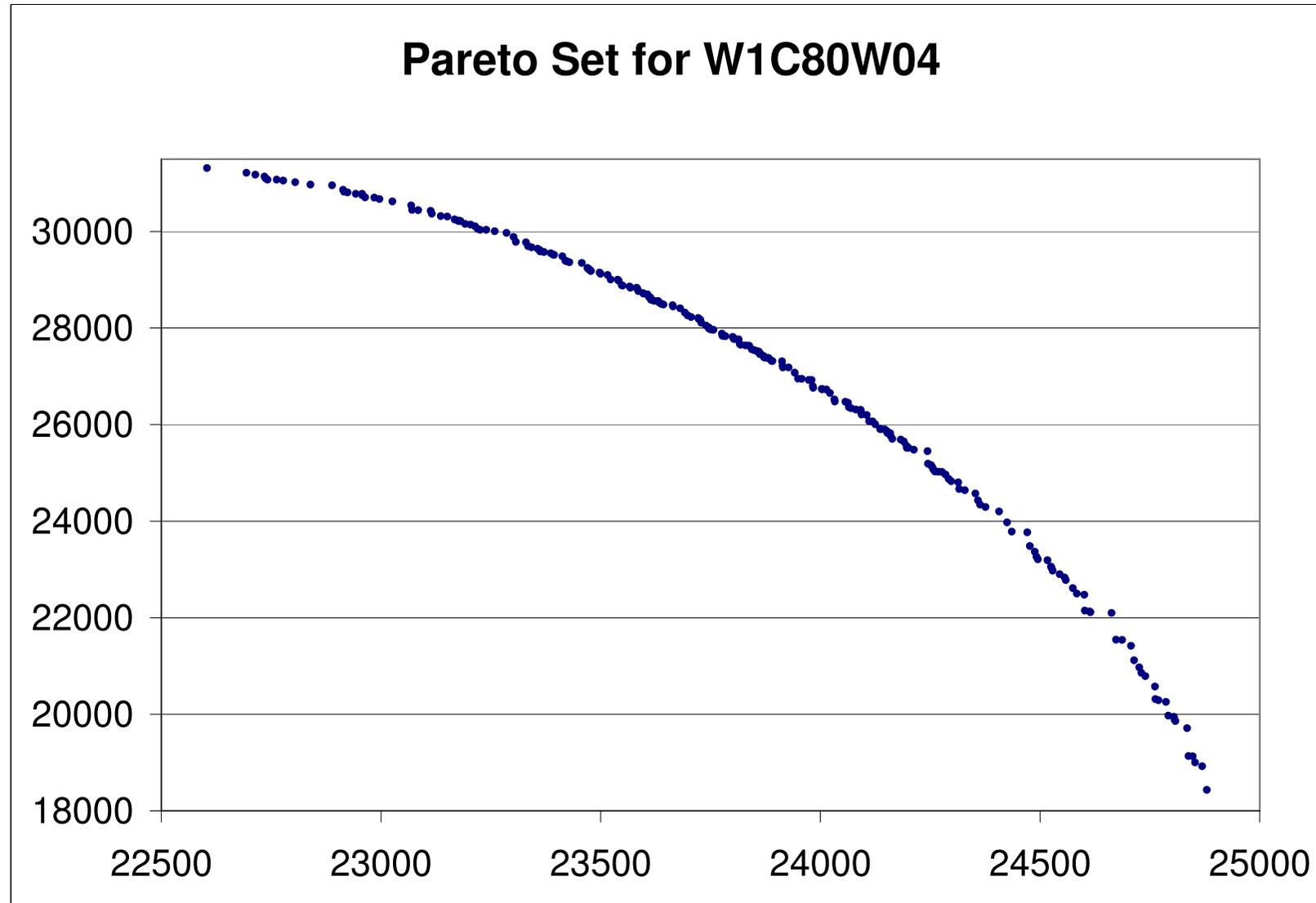
Implementing the Parallel WCN Algorithm

- The algorithm was parallelized using **MWBlackBox**, a tool for deploying simple master-worker algorithms on the computational grid.
- **MWBlackBox** is built on top of **Condor**, a unique full-featured task management system.
- **Condor** is used to remotely run a subproblem solver implemented using the **SYMPHONY** callable library.
- Required methods
 - `get_userinfo()`: Specify file locations
 - `setup_initial_tasks()`: Find utopia point
 - `act_on_completed_task()`: Generate new subproblems
 - `printresults()`: Print final results

Scalability Issues

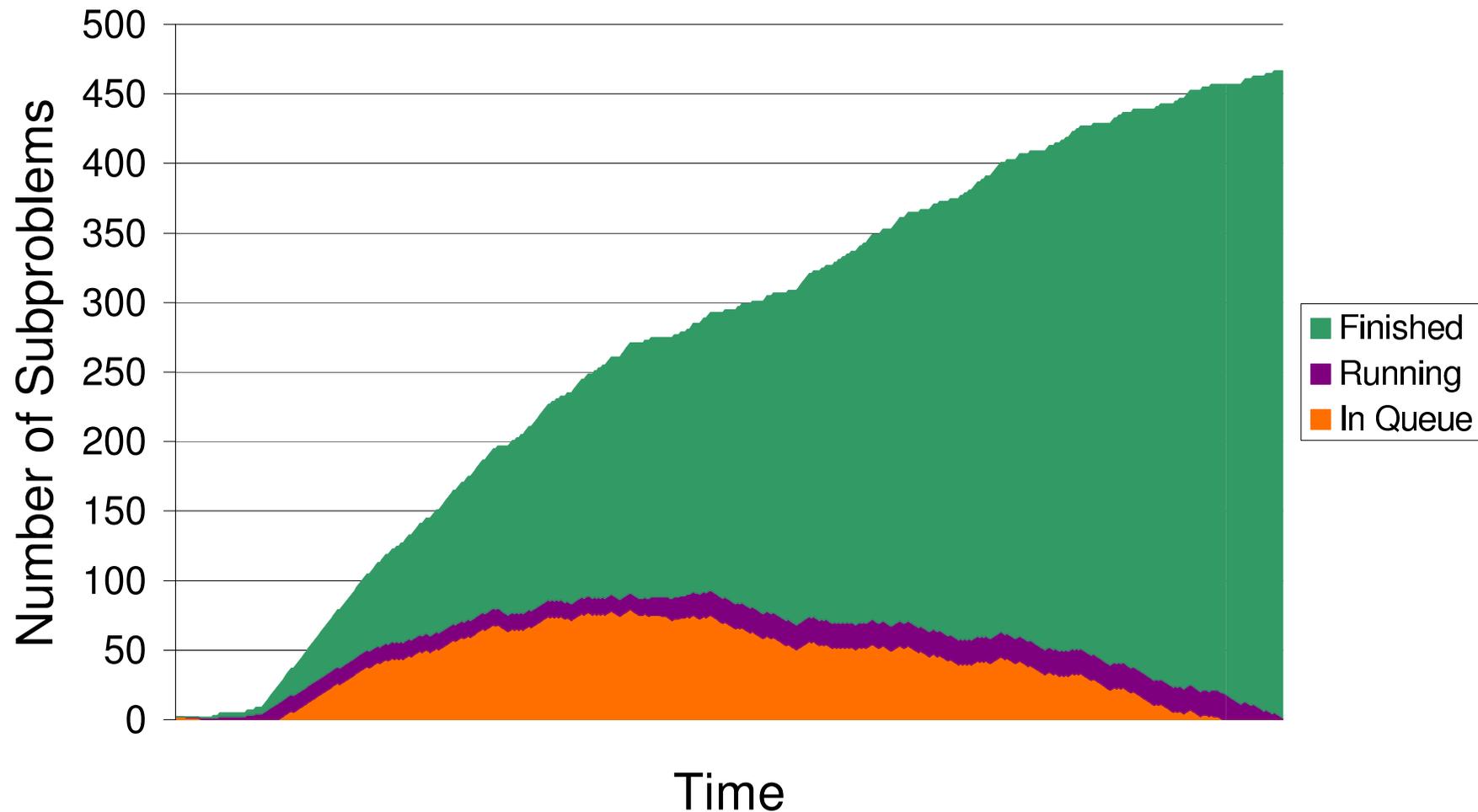
- The scalability issues are very similar to **parallel branch and bound**.
 - There is a queue of independent tasks to be done.
 - Each task may generate two child tasks, but there is no way of knowing a priori what the tree of tasks will look like.
 - The order of processing the tasks does not matter for correctness, but can greatly affect parallel performance.
- The main scalability factors
 - **The number of outcomes** and their distribution.
 - **How fast the queue grows** in the beginning and shrinks at the end.
 - If **warm starting** or a **global cut pool** is used, the **processing order** may also affect subproblem solution time.
- To test scalability of the basic algorithm, we solved 32 instances of the multicriteria knapsack problem with different numbers of available processors.

Example: Pareto Set for W1C80W04

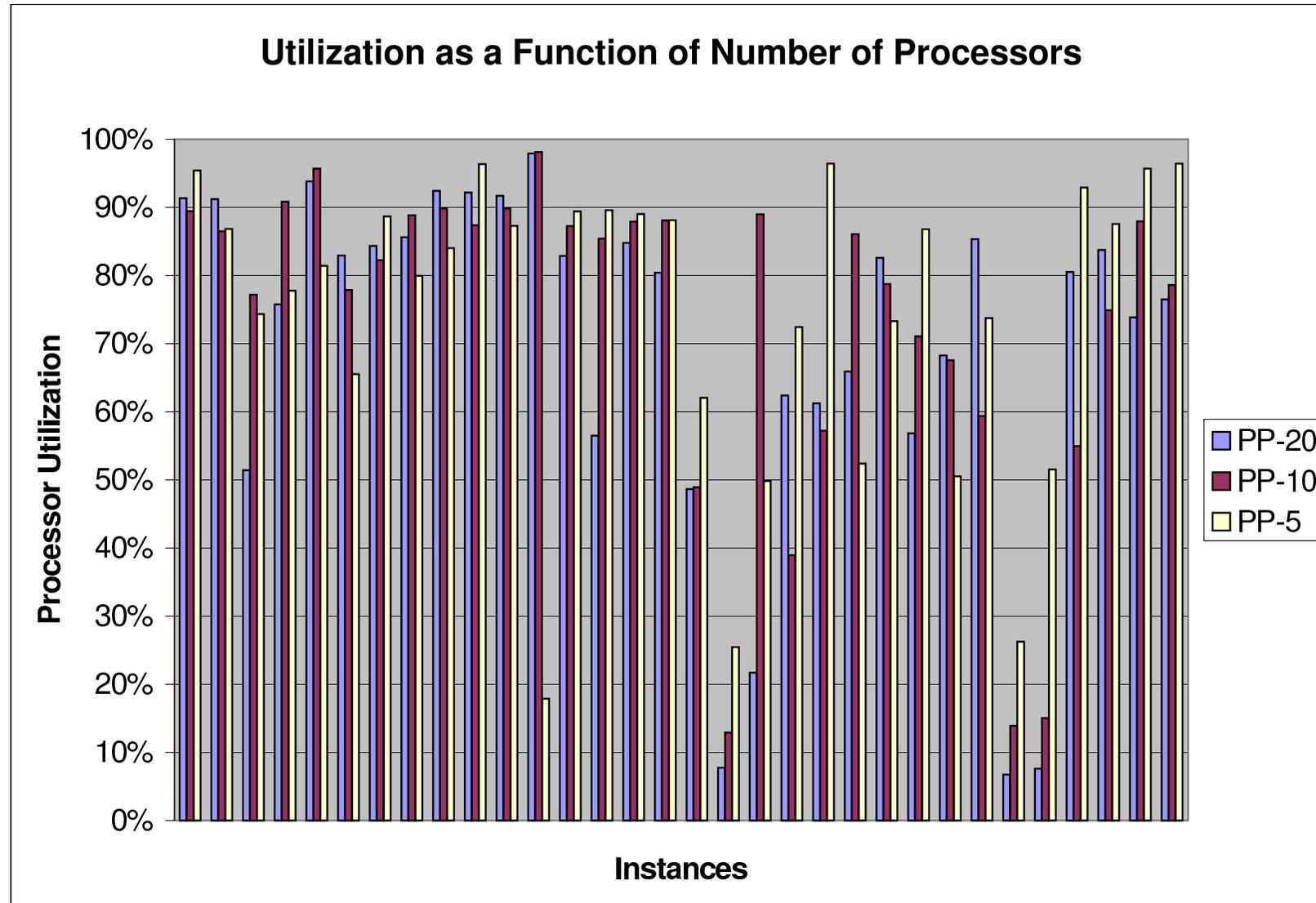


Example: Queue Size for W1C80W04

Evolution of Subproblem Queue



Computational Results: Processor Utilization



Future Work: Improving Parallel Performance

- Limiting **ramp-up** and **ramp-down** time
 - Solution of subproblems can itself be parallelized when the queue is small.
 - Searching the widest intervals first may help populate the queue more quickly.
 - Subintervals could be allocated to processors a priori without solving any initial subproblems
- More **asynchronicity** can be introduced by allowing each worker to search an entire interval recursively.
- Maintaining **warm starting** information
 - For very large instances, warm starting can help a lot.
 - However, this means the subproblem descriptions will become much larger.
 - One option is to store the warm starts locally.
- Cuts can be shared through the use of a **global cut pool**.

Conclusion

- Generating the complete set of Pareto outcomes is a challenging computational problem.
- We presented a new algorithm for solving biobjective mixed-integer programs.
- The algorithm is
 - asymptotically optimal,
 - generates exact breakpoints,
 - has good numerical properties, and
 - can exploits modern solution techniques.
- We have shown how this algorithm is implemented in the SYMPHONY MILP solver framework.
- Future work
 - Improvements to warm starting procedures
 - Improvements to the parallelization scheme
 - More than two objective

More Information

- SYMPHONY

- Prepackaged releases can be obtained from www.BranchAndCut.org.
- Up-to-date source is available from www.coin-or.org.
- Available Solvers

- Generic MILP
- Traveling Salesman Problem
- Vehicle Routing Problem
- Mixed Postman Problem
- Biobjective Knapsack Solver
- Set Partitioning Problem
- Matching Problem
- Network Routing

- For references and further details, see *An Improved Algorithm for Biobjective Integer Programming*, to appear in *Annals of OR*, available from

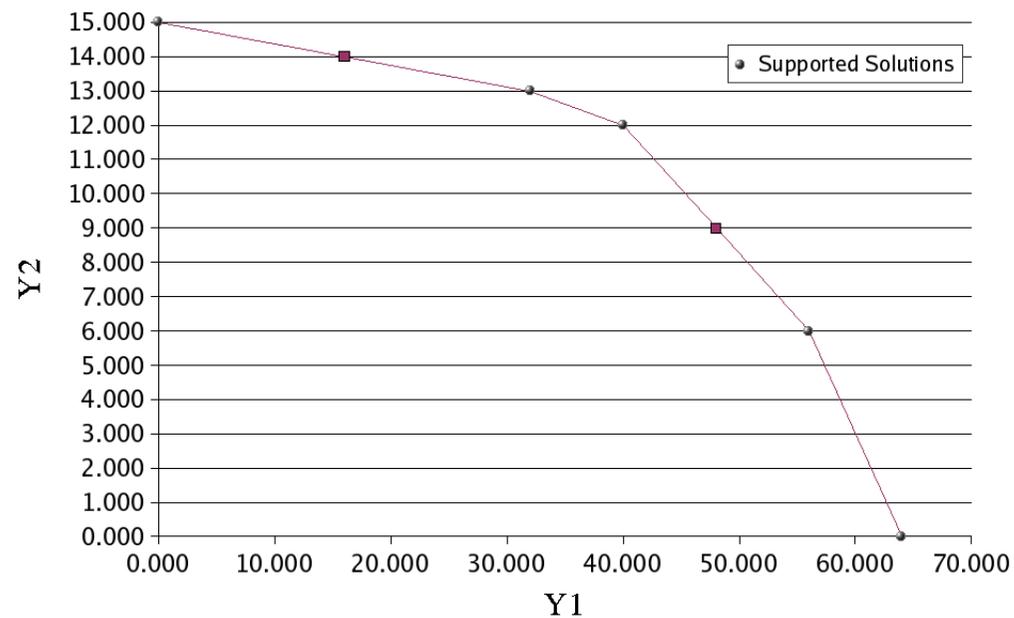
www.lehigh.edu/~tkr2

- Overviews of multiobjective integer programming
 - Climaco (1997)
 - Ehrgott and Gandibleux (2002)
 - Ehrgott and Wiecek (2005)

Quick Example

$$\begin{aligned} \text{vmax} \quad & [8x_1, x_2] \\ \text{s.t.} \quad & 7x_1 + x_2 \leq 56 \\ & 28x_1 + 9x_2 \leq 252 \\ & 3x_1 + 7x_2 \leq 105 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Non-dominated Solutions



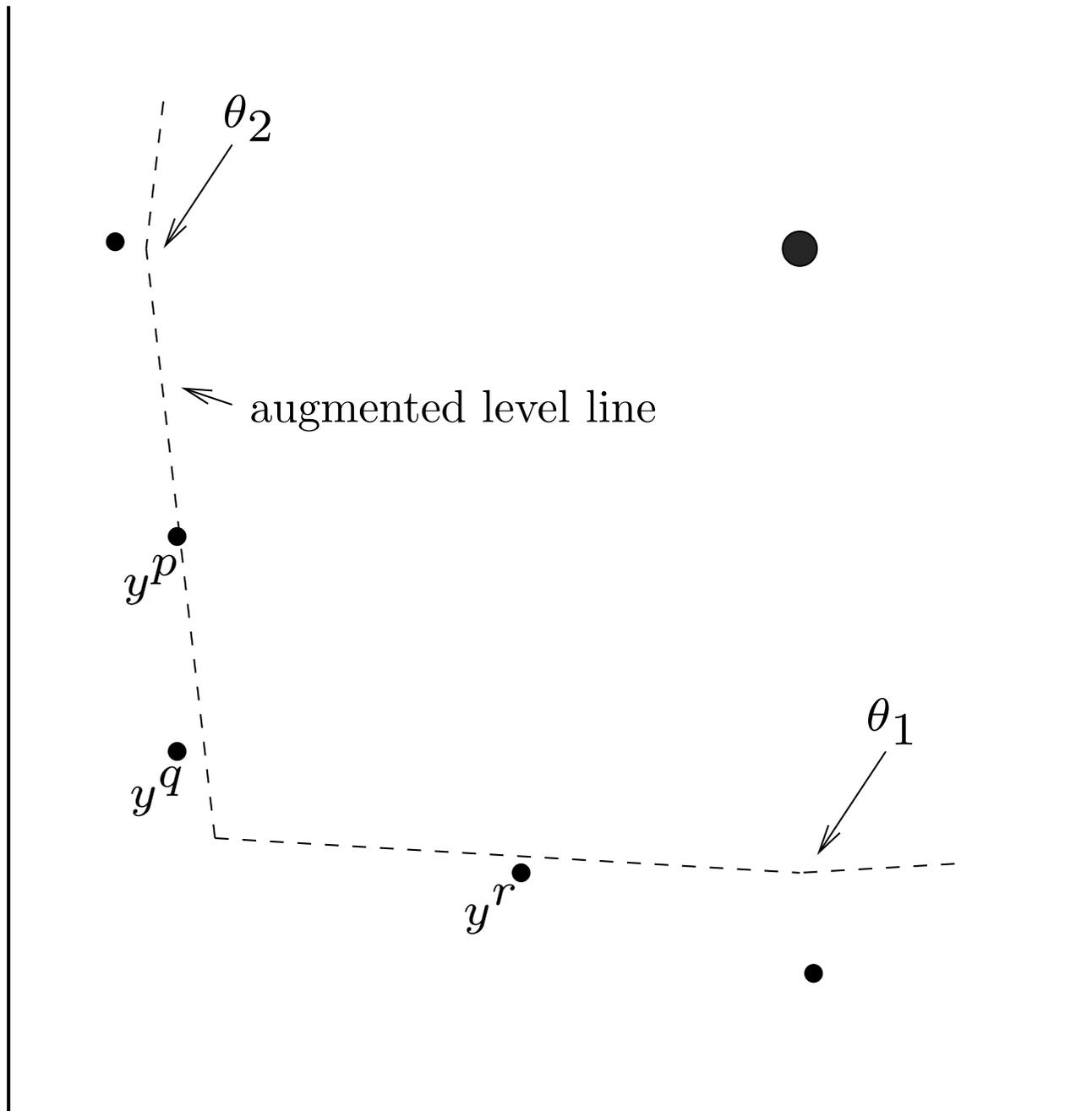
Relaxing the Uniform Dominance Requirement

- Dealing with weakly dominated outcomes is the most challenging aspect of these methods.
- We need a method of preventing $P(\beta)$ from producing weakly dominated outcomes.
- Weakly dominated outcomes are the same WCN distance from the ideal point as the outcomes they are dominated by.
- However, they are farther from the ideal point as measured by the l_p norm for $p < \infty$.
- One solution is to replace the WCN with the *augmented Chebyshev norm* (ACN), defined by

$$\|(y_1, y_2)\|_{\infty}^{\beta, \rho} = \max\{\beta|y_1|, (1 - \beta)|y_2|\} + \rho(|y_1| + |y_2|),$$

where ρ is a small positive number.

Illustrating the ACN



Solving $P(\beta)$ with the ACN

- The problem of determining the outcome closest to the ideal point under this metric is

$$\begin{array}{ll}
 \min & z + \rho(|y_1^* - y_1| + |y_2^* - y_2|) \\
 \text{subject to} & z \geq \beta(y_1^* - y_1) \\
 & z \geq (1 - \beta)(y_2^* - y_2) \\
 & y \in f(X).
 \end{array} \tag{4}$$

- Because $y_k^* - y_k \geq 0$ for all $y \in f(X)$, the objective function can be rewritten as

$$\min z - \rho(y_1 + y_2).$$

- For fixed $\rho > 0$ small enough:
 - all optimal outcomes for problem (4) are Pareto (in particular, they are not weakly dominated), and
 - for a given Pareto outcome y for problem (4), there exists $0 \leq \hat{\beta} \leq 1$ such that y is the unique outcome to problem (4) with $\beta = \hat{\beta}$.
- In practice, choosing a proper value for ρ can be problematic.

Combinatorial Method for Eliminating Weakly Dominated Solutions

- In the case of *biobjective linear integer programs* (BLIPs), we can employ combinatorial methods.
- Such a strategy involves implicitly enumerating alternative optimal solutions to $P(\beta)$.
- Weakly dominated outcomes are eliminated with cutting planes during the branch and bound procedure.
- Instead of pruning nodes that yield feasible outcomes immediately, we continue to search for alternative optima that dominate the current incumbent.
- To do so, we determine which of the two constraints

$$z \geq \beta(y_1^* - y_1)$$

$$z \geq (1 - \beta)(y_2^* - y_2)$$

from problem (1) is binding at \hat{y} .

Combinatorial Method for Eliminating Weakly Dominated Solutions (cont'd)

- Let ϵ_1 and ϵ_2 be such that if y_r is a new outcome between y^p and y^q , then $y_i^r \geq \min\{y_i^p, y_i^q\} + \epsilon_i$, for $i = 1, 2$.
- If the first constraint is binding, then the cut

$$y_1 \geq \hat{y}_1 + \epsilon_1$$

is valid for any outcome that dominates \hat{y} .

- If the second constraint is binding, then the cut

$$y_2 \geq \hat{y}_2 + \epsilon_2$$

is valid for any outcome that dominates \hat{y} .

Hybrid Methods

- In practice, the ACN method is fast, but choosing the proper value of ρ is problematic.
- Combinatorial methods are less susceptible to numerical difficulties, but are slower.
- Combining the two methods improves running times and reduces dependence on the magnitude of ρ .

Other Enhancements to the Algorithm

- In Step 2, any new outcome y^r will have $y_1^r > y_1^p$ and $y_2^r > y_2^q$.
- If no such outcome exists, then the subproblem solver must still re-prove the optimality of y^p or y^q .
- Then it must be the case that

$$\|y^* - y^r\|_{\infty}^{\beta_{pq}} + \min\{\beta_{pq}\epsilon_1, (1 - \beta_{pq})\epsilon_2\} \leq \|y^* - y^p\|_{\infty}^{\beta_{pq}} = \|y^* - y^q\|_{\infty}^{\beta_{pq}}$$

- Hence, we can impose an a priori upper bound of

$$\|y^* - y^p\|_{\infty}^{\beta_{pq}} - \min\{\beta_{pq}\epsilon_1, (1 - \beta_{pq})\epsilon_2\}$$

when solving the subproblem $P(\beta_{pq})$.

- With this upper bound, each subproblem will either be infeasible or produce a new outcome.

Using Warm Starting

- We have been developing methodology for *warm starting* branch and bound computations.
- Because the WCN algorithm involves solving a sequence of slightly modified MILPs, warm starting can be used.
- **Three approaches**
 - Warm start from the result of the previous iteration.
 - Solve a “base” problem first and warm each subsequent problem from there.
 - Warm start from the “closest” previously solved subproblem.
- In addition, we can optionally save the global cut pool from iteration to iteration.

Implementation: Code Sample

- Recall the example from earlier:

$$\begin{aligned} \text{vmax} \quad & [8x_1, x_2] \\ \text{s.t.} \quad & 7x_1 + x_2 \leq 56 \\ & 28x_1 + 9x_2 \leq 252 \\ & 3x_1 + 7x_2 \leq 105 \\ & x_1, x_2 \geq 0 \end{aligned}$$

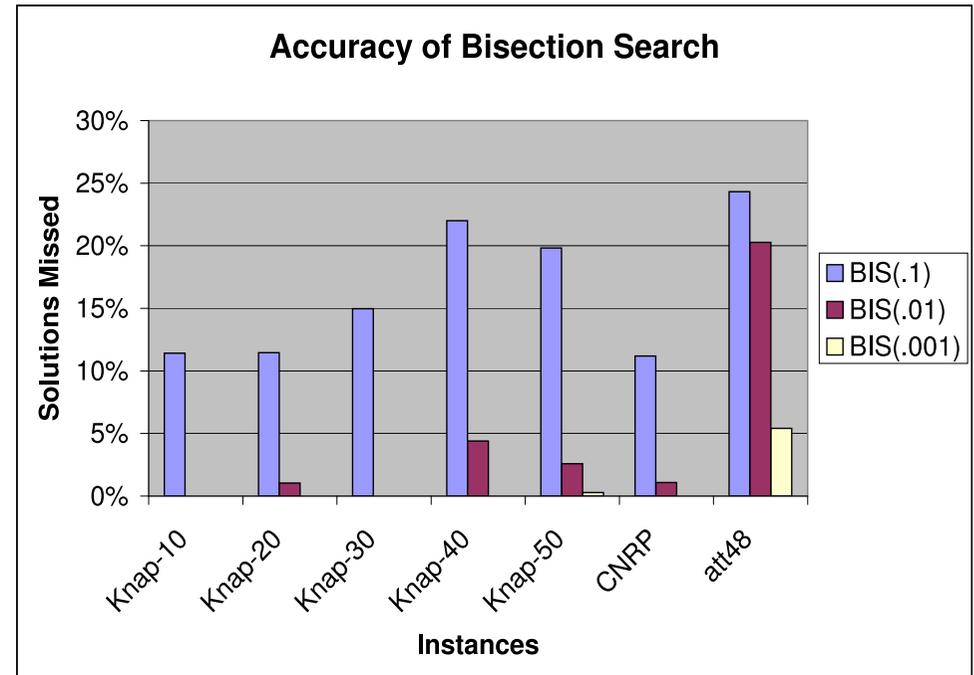
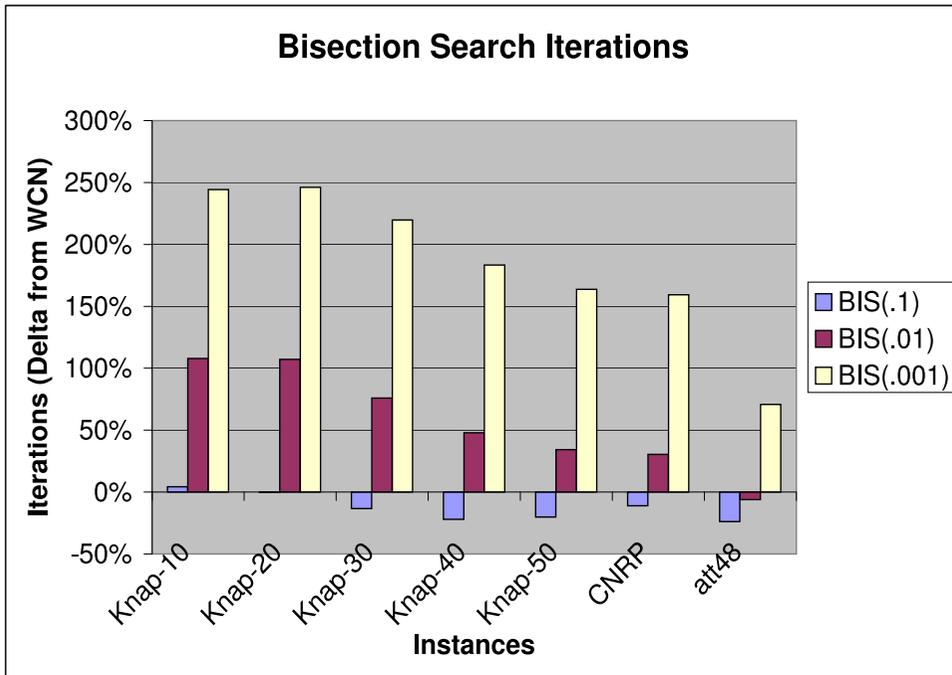
- The following code solves this model using **SYMPHONY**.

```
int main(int argc, char **argv)
{
    OsiSymSolverInterface si;
    si.parseCommandLine(argc, argv);
    si.setObj2Coeff(1, 1);
    si.loadProblem();
    si.multiCriteriaBranchAndBound();
}
```

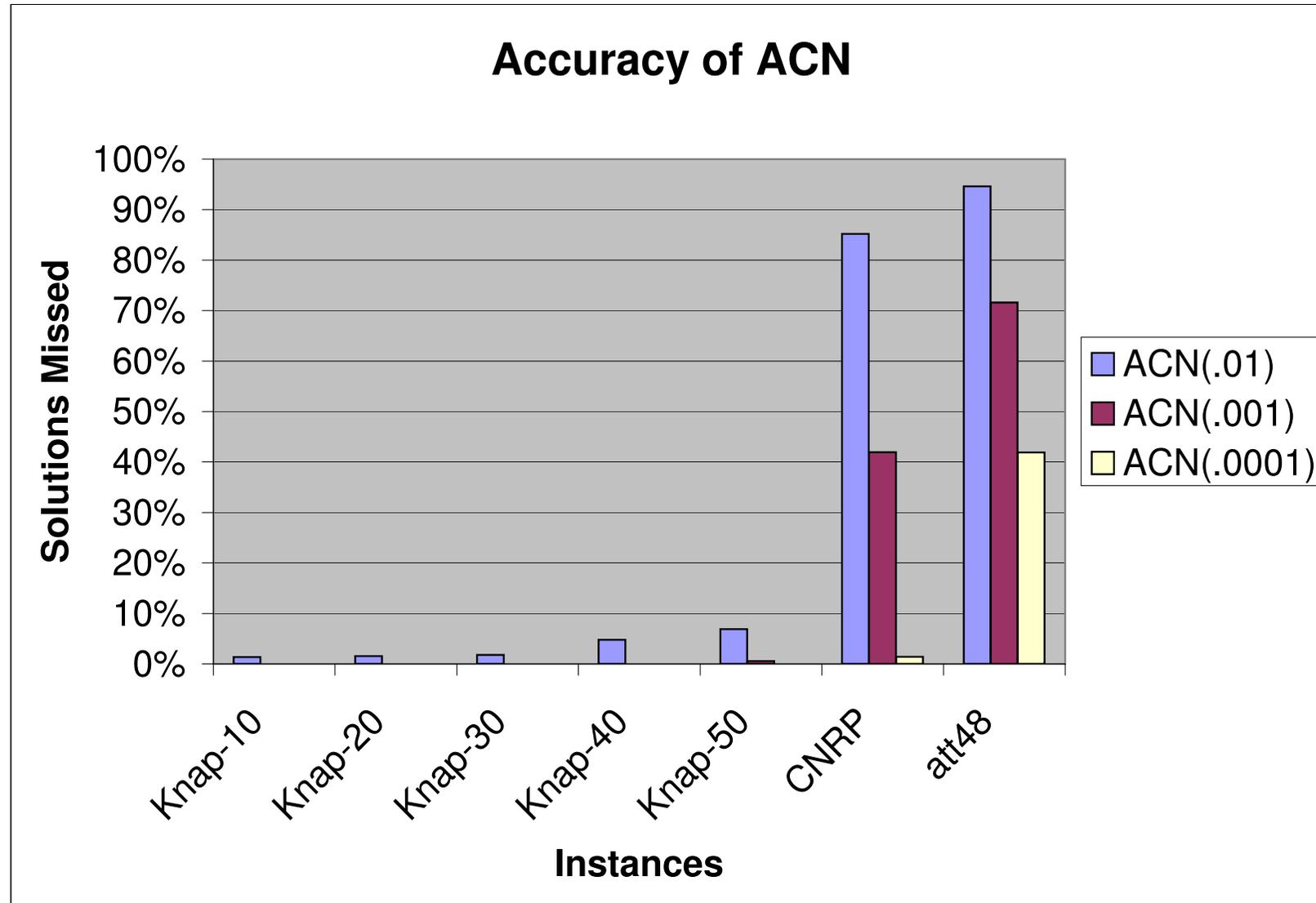
Uniform Dominance

- Members of X that are *not* strongly dominated by some efficient solution are called *weakly dominated*.
- Weakly dominated solutions are optimal to (1) for some β .
- If X does not contain any weakly dominated solutions, then the instance is said to be *uniformly dominant*.
- The assumption of uniform dominance simplifies computation substantially, but is not satisfied in most practical settings.
- The deal with this, we need to modify the algorithm.

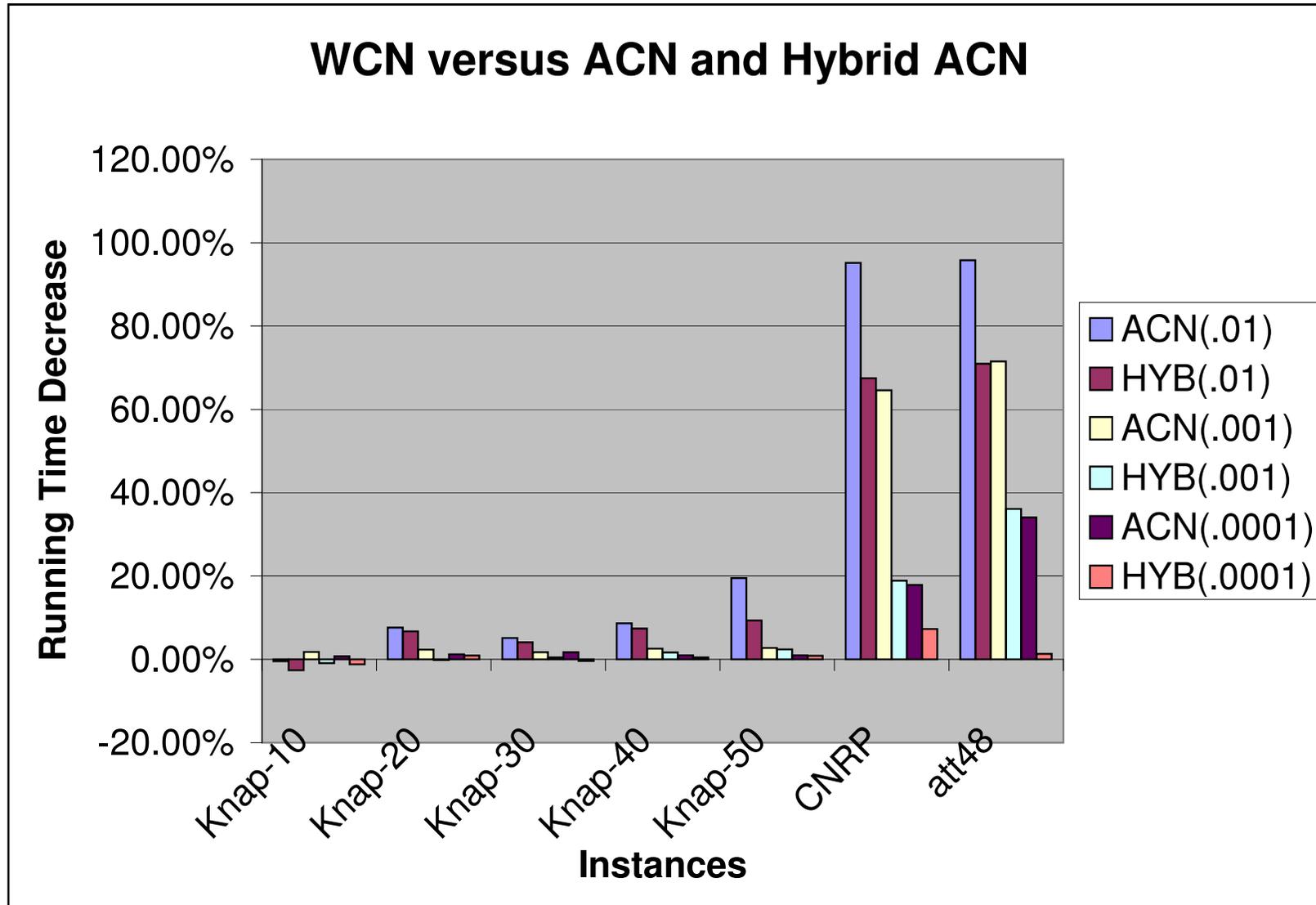
Computational Results: WCN versus Bisection Search



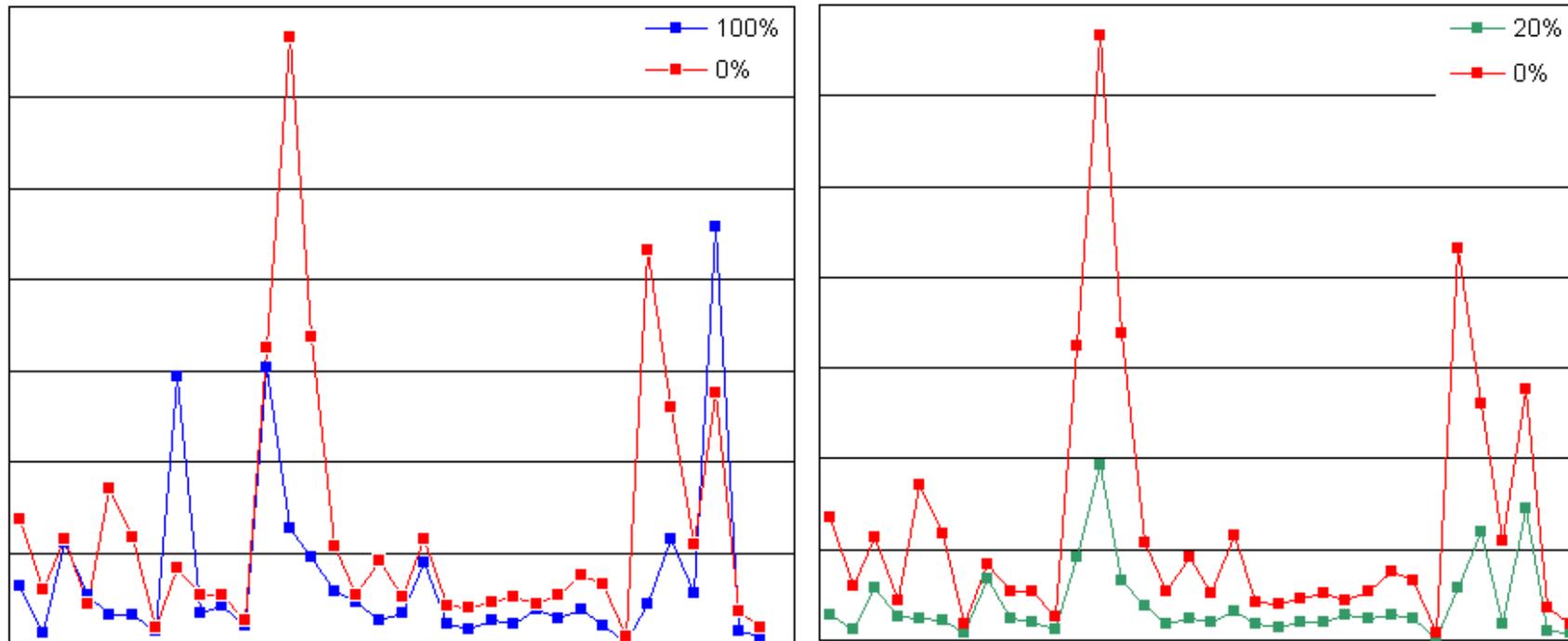
Computational Results: Accuracy of ACN



Computational Results: Running Time Comparison



Computational Results: Using Warm Starting to Solve CNRP Instances



These are results using **SYMPHONY** to solve CNRP instances with two different warm starting strategies.