Rapid Implementation of Branch-and-Cut with Heuristics using GAMS

Herman Abeledo and Hua (Edward) Ni
George Washington University

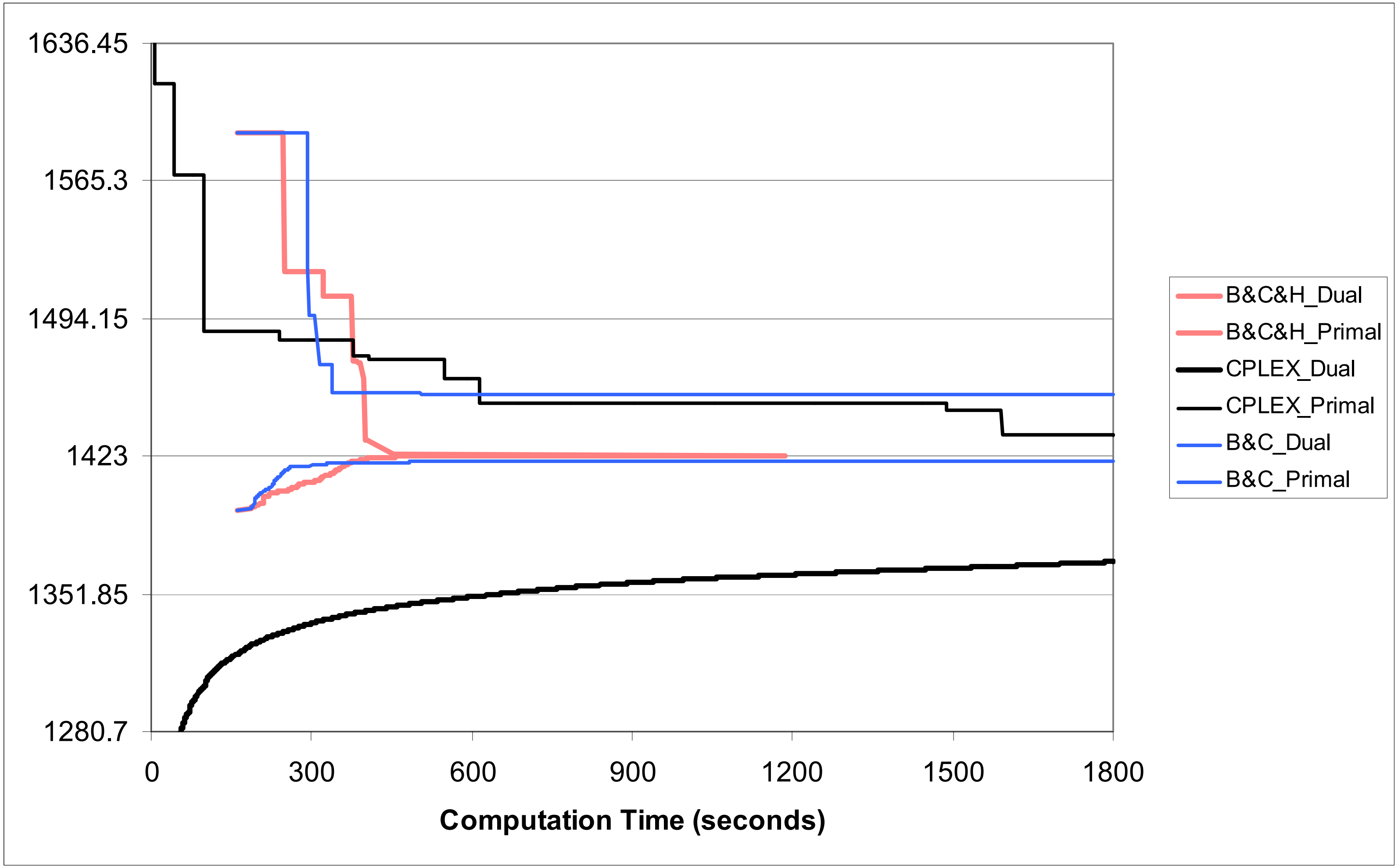
Michael Bussieck and Alex Meeraus
GAMS Development Corporation

Informs Annual Meeting 2003
October 19-22, 2003, Atlanta, Georgia

Branch-and-Cut

- Branch-and-cut is an established algorithm to improve the branch-and-bound search.
- Implementation facilities:
  - MIP solver callback functions (e.g. CPLEX, XPRESS)
  - Branch-and-cut framework (e.g. ABACUS, SYMPHONY, COIN BCP)
- Heuristic algorithms can help branch-and-bound search by strong pruning.

Convergence – Pipeline Design



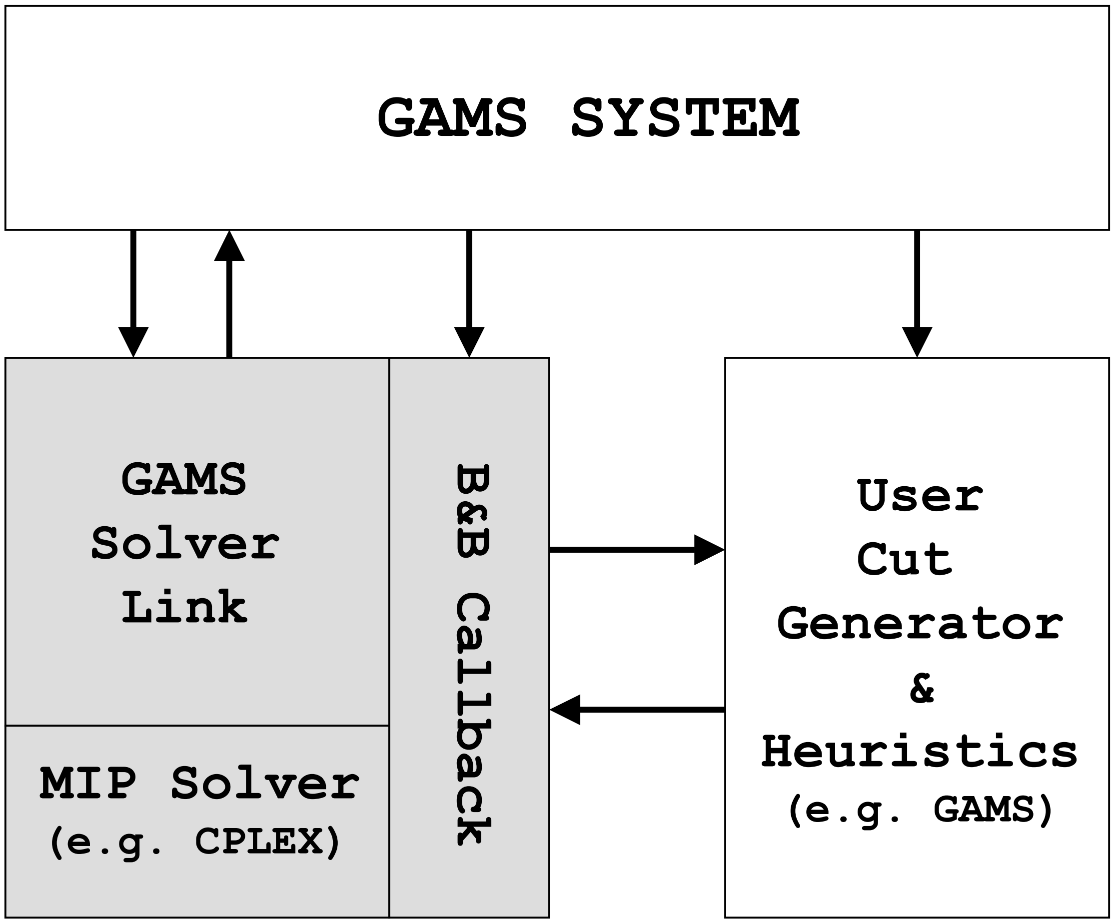
Knowledge Requirements

- IT knowledge (programming in C/C++, JAVA, Solver APIs)
- Mathematical programming knowledge
- Application specific knowledge

Design Goal

- Supply GAMS users with an easy access to Branch-and-cut capabilities.
- Relieve GAMS users from the burden of learning a new programming language and/or acquiring solver-specific knowledge, so that they can focus more on the design of cutting planes and heuristic algorithms.

Design Principle



Implementation Issues

- GDX interface for information exchange
- Namespace mapping facility
- Option file support
- DLL/SO or GAMS support
- Incumbent solution validation

A Steiner Tree Problem

- Ortega, F., Wolsey, L. 2003. A Branch-and-cut Algorithm for the Single-commodity, Uncapacitated, Fixed-charge Network Flow Problem. Networks, Vol 41, 143-158
- Berlin52 – from SteinLib
  – 52 nodes (16 terminals)
  – 1326 edges

Model Formulation

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} (c_{ij}x_{ij} + f_{ij}y_{ij}) \\
\text{s.t.} \quad & \sum_{j\in V_i^-} x_{ji} - \sum_{j\in V_i^+} x_{ij} = b_i \\
& x_{ij} \leq U y_{ij} \quad \forall\ (i,j)\in A \\
& x_{ij} \geq 0 \quad \forall\ (i,j)\in A \\
& y_{ij} \in \{0,1\} \quad \forall\ (i,j)\in A
\end{aligned}
$$

Dicut Inequality and Separation

- Dicut inequality:

$$\sum_{(i,j)\in\delta^-(S)} y_{ij} \geq 1$$

if $S \subset V$ and $b(S)>0$.

- Separation:

$$\xi = \min\left\{ \sum_{(i,j)\in A} \bar{y}_{ij} z_j (1 - z_i) \ : \ \sum_{i\in V} b_i z_i > 0, \ z_i \in \{0,1\} \ \forall \ i \in V \right\}$$

A Simple Heuristic

- Delete the arcs with zero values in the current fractional solution.
- Solve the original model in a smaller scale.

Cut Generator in GAMS I

C:\Documents and Settings\Administrator\Desktop\Dicuts.gms

Dicuts.gms

```
Set nn          nodes
    arc(nn,nn)  arcs; alias(nn,n,m);
Parameter demand(nn)       node demand
          fcost(nn,nn)     fixed cost
$gdxin NetInfo.gdx
$load nn demand fcost

Variables ybar(nn,nn)     the fractional y solution
          cost
binary variables z(nn);
Equations Obj       objective
          SC        positive demand over the node block;
Obj..     sum(arc(m,n), ybar.l(m,n)*z(n)*(1-z(m))) =e= cost;
SC..      sum(n, demand(n)*z(n)) =g= 1;

Model Dicut /Obj, SC/;
execute_load 'CutCBSol.gdx' ybar=y;
arc(m,n)$fcost(m,n) = yes;
Solve Dicut mini cost using MINLP;
```

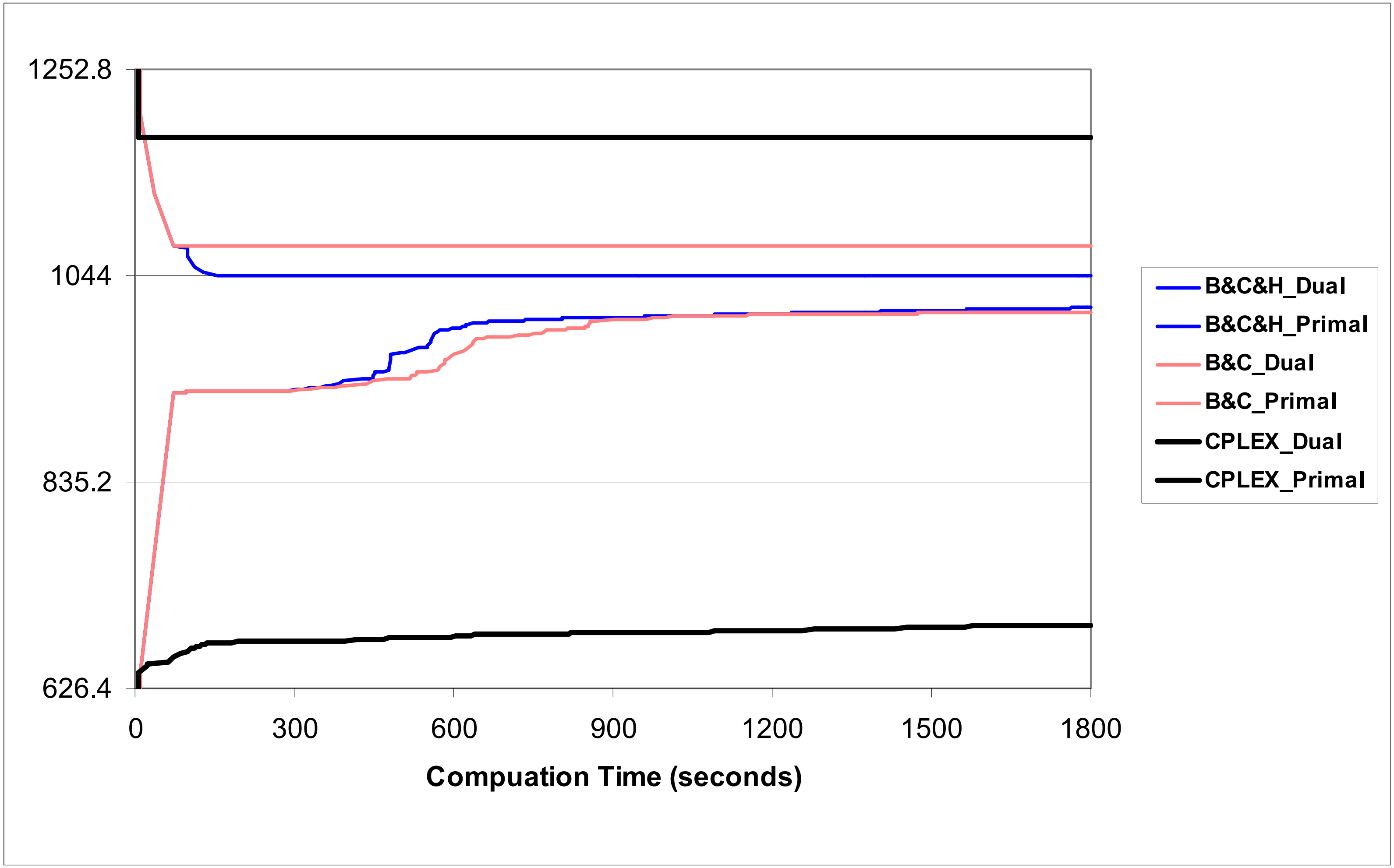
Cut Generator in GAMS II

C:\Documents and Settings\Administrator\Desktop\Dicuts.gms

Dicuts.gms

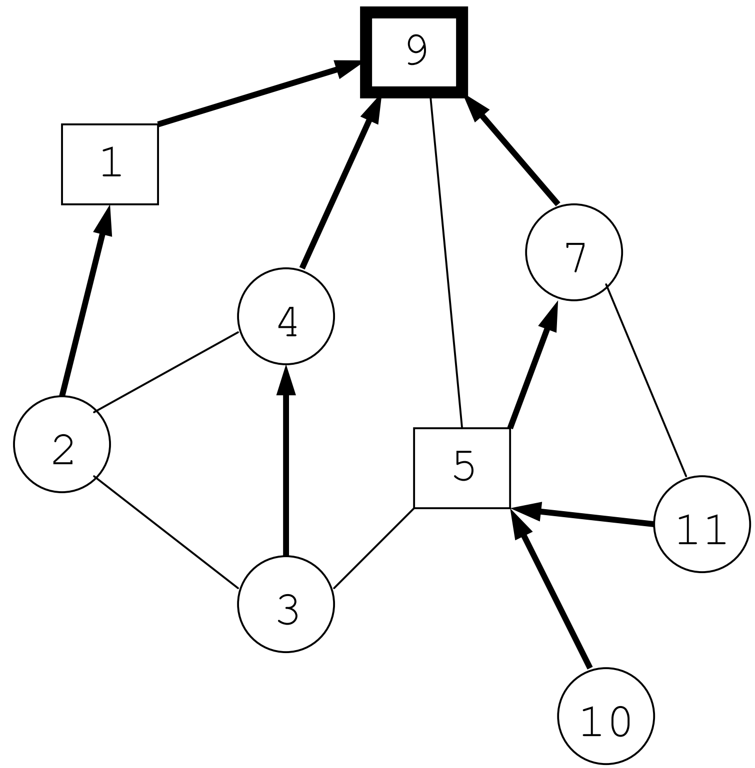
```
arc(m,n)$fcost(m,n) = yes;
Solve Dicut mini cost using MINLP;

Set S(nn) Nodes in the node block; S(n) = round(z.l(n));
Set cc number of cuts generated  / 1 /;

Parameter CC_y(cc,nn,nn)          coefficient of the y variables in the cut
          CRHS(cc)                cut rhs
          CSENSE(cc)              the sense of the cuts
          NUMCUTS /1/;

if(cost.l>=1, NUMCUTS = 0;
else CSENSE(cc) = 3;
     CRHS(cc)   = 1;
     CC_y(cc,arc(m,n)) = not S(m) and S(n);
);

execute_unload 'CutCBCuts.gdx' NUMCUTS, CRHS, CSENSE, CC_y ;
```

Convergence – Steiner Tree



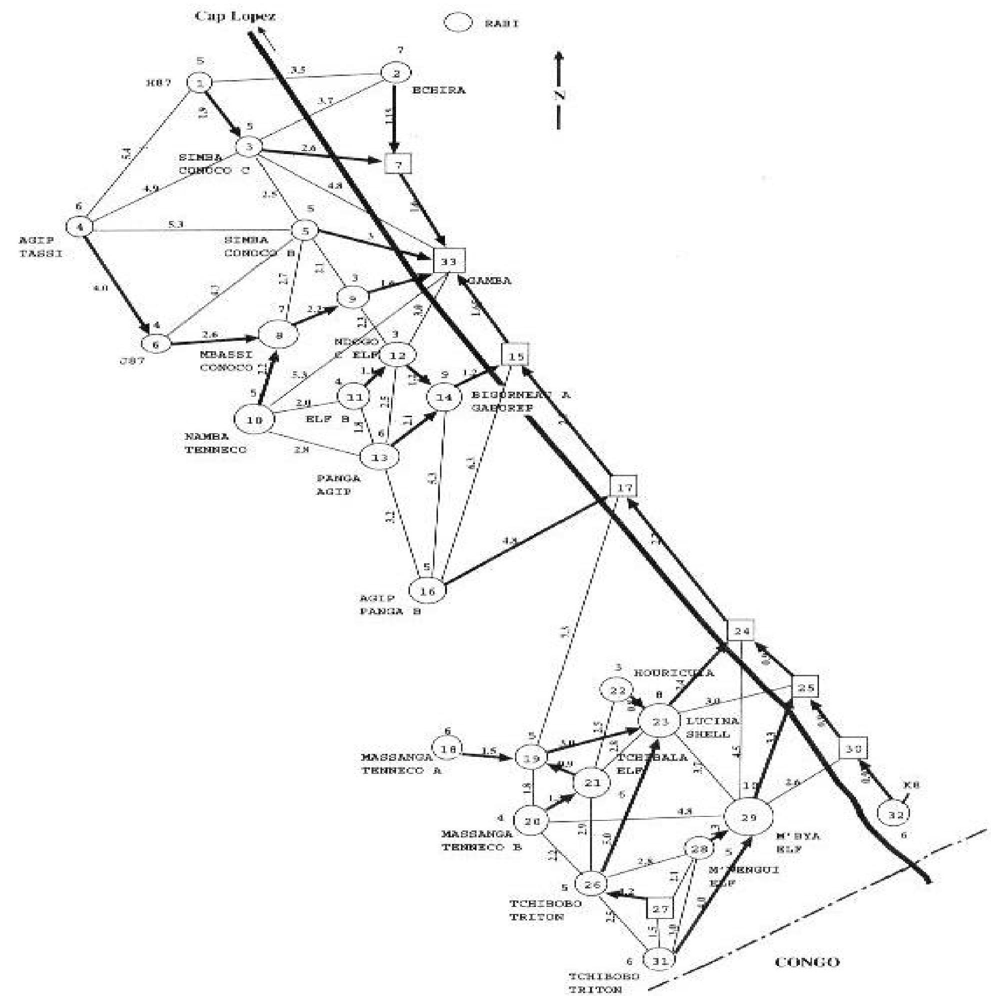
An Oil Pipeline Design Problem

- Brimberg, J., et. al. 2003. An Oil Pipeline Design Problem. *Operations Research*, Vol 51, 228-239



South Gabon Oil field

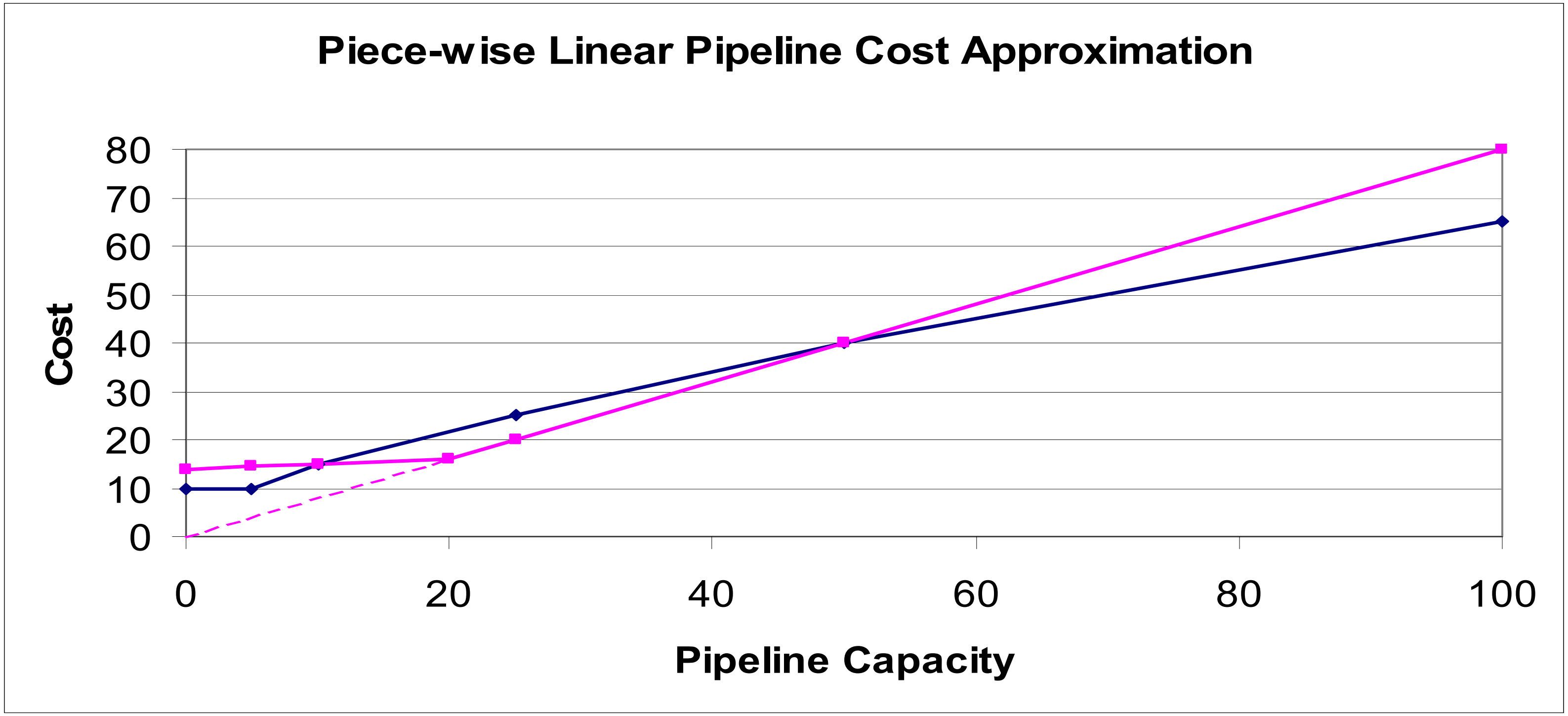
- 33 nodes
  - 25 wells
  - 7 connections
  - 1 port
- 129 arcs
- 5 types of pipe



Model Formulation

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} \sum_{k\in\hat{K}} E_{ij}^k y_{ij}^k \\
\text{s.t.} \quad & \sum_{j\in S(i)} \sum_{k\in\hat{K}} y_{ij}^k = 1 \quad \forall\, i \in N \setminus \{n\} \\
& \sum_{j\in S(i)} f_{ij} - \sum_{j\in P(i)} f_{ji} = p_i \quad \forall\, i \in N \setminus \{n\} \\
& f_{ij} - \sum_{k\in\hat{K}} C^k y_{ij}^k \leq 0 \quad \forall\, (i,j) \in A \\
& f_{ij} \geq 0 \quad \forall\, (i,j) \in A \\
& y_{ij}^k \in \{0,1\} \quad \forall\, (i,j) \in A,\ k \in K
\end{aligned}
$$

Cost Approximation



	Type 1	Type 2	Type 3	Type 4	Type 5
Capacity	5	10	25	50	100
Cost	10	15	25	40	65
Approx. Cost	14.5	15	20	40	80

Two-Stage Heuristic Algorithm

- Stage One: Piecewise linear cost approximation

$$\min \sum_{(i,j)\in A} (l_{ij}(\alpha f_{ij} + (\alpha - \beta) s_{ij}))$$

$$\begin{aligned}
\text{s.t.} \quad \sum_{j\in S(i)} z_{ij} &\leq 1 \quad \forall\, i \in N \\
\sum_{j\in S(i)} f_{ij} - \sum_{j\in P(i)} f_{ji} &= p_i \quad \forall\, i \in N \setminus \{n\} \\
f_{ij} - C_{\max} z_{ij} &\leq 0 \quad \forall\, (i,j) \in A \\
f_{ij} + s_{ij} - h z_{ij} &\geq 0 \quad \forall\, (i,j) \in A \\
f_{ij} &\geq 0 \quad \forall\, (i,j) \in A \\
s_{ij} &\geq 0 \quad \forall\, (i,j) \in A \\
z_{ij} &\in \{0,1\}\ \forall (i,j) \in A.
\end{aligned}$$

- Stage Two: Layout fixing

Cutting Plane Generation

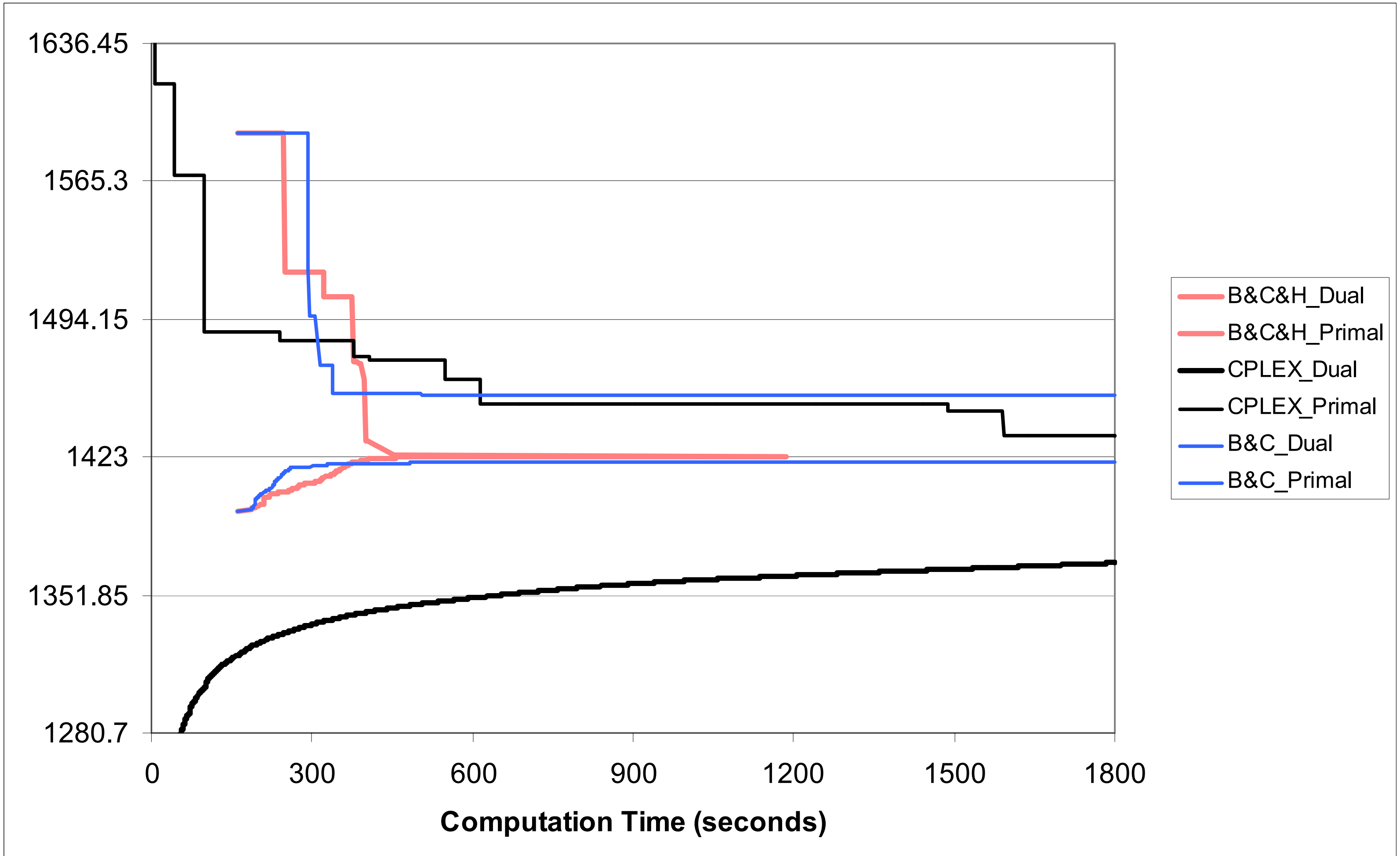
- Valid cut inequality

$$\sum_{i\in W}\sum_{j\in N}\sum_{k\in K} E_{ij}^k y_{ij}^k \geq V_{\mathrm{st}}$$

- Generation

$$W_i = \{j \mid j = i, \text{ or } j \text{ is a descendant of } i\}$$

Convergence – Pipeline Design



Computational Results

- Overhead
  - Time spent within the callback functions minus MIP computation on cuts and heuristics.
  - 20% ~ 25%
- Performance Improvements
  - Steiner: 6 hours vs. 2+ days
  - Pipeline Design: 20 minutes vs. 450 minutes