Lattices and Integer Optimization - A Tutorial (Part II)

Bala Krishnamoorthy
Department of Mathematics, WSU

April 29, 2010

Basis Reduction (BR) and Discrete Optimization

- Lenstra (1983) - poly-time algo for integer programming (IP) in fixed dimensions (also, Kannan (1985))
  - BR is one of the key steps

Basis Reduction (BR) and Discrete Optimization

- Lenstra (1983) - poly-time algo for integer programming (IP) in fixed dimensions (also, Kannan (1985))
  - BR is one of the key steps
  - not implemented in practice

Basis Reduction (BR) and Discrete Optimization

- Lenstra (1983) - poly-time algo for integer programming (IP) in fixed dimensions (also, Kannan (1985))
  - BR is one of the key steps
  - not implemented in practice
- more straightforward application of BR to IP
  - column basis reduction

Basis Reduction (BR) and Discrete Optimization

- Lenstra (1983) - poly-time algo for integer programming (IP) in fixed dimensions (also, Kannan (1985))
  - BR is one of the key steps
  - not implemented in practice

- more straightforward application of BR to IP
  - column basis reduction (joint work with Pataki (UNC))
  - simple; works in practice
  - theory for class of knapsack problems,
    and for general IPs (Pataki et al., 2010)

Basis Reduction (BR) and Discrete Optimization

- Lenstra (1983) - poly-time algo for integer programming (IP) in fixed dimensions (also, Kannan (1985))
  - BR is one of the key steps
  - not implemented in practice

- more straightforward application of BR to IP
  - column basis reduction (joint work with Pataki (UNC))
  - simple; works in practice
  - theory for class of knapsack problems, and for general IPs (Pataki et al., 2010)

- lattice-based approaches to number partitioning in hard phase (joint work with Bill Webb, Nathan Moyer (WSU))

Integer Programming (IP)

- IP feasibility

  Given

$$P \; = \; \{ \mathbf{x} \,|\, \boldsymbol{\ell} \leq A\mathbf{x} \leq \mathbf{b} \},$$

  Find $\mathbf{x} \in P \cap \mathbb{Z}^n$, or prove that no such $\mathbf{x}$ exists.

Branching for IP feasibility

Given polyhedron P, integral vector $\mathbf{c}$,

- $\mathrm{width}(\mathbf{c}, P) = \max\{\, \mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P \,\} - \min\{\, \mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P \,\} = \gamma - \delta.$

Branching for IP feasibility

Given polyhedron P, integral vector $\mathbf{c}$,

- $\mathrm{width}(\mathbf{c}, P) = \max\{\, \mathbf{c}^T\mathbf{x} \,|\, \mathbf{x} \in P \,\} - \min\{\, \mathbf{c}^T\mathbf{x} \,|\, \mathbf{x} \in P \,\} = \gamma - \delta.$

- **branching on $\mathbf{c}^T\mathbf{x}$** means creating the branches $\mathbf{c}^T\mathbf{x} = \lceil \delta \rceil$, $\mathbf{c}^T\mathbf{x} = \lceil \delta \rceil + 1, \ldots, \mathbf{cx} = \lfloor \gamma \rfloor$ (add constraint to LP relaxation).

Branching for IP feasibility

Given polyhedron P, integral vector $\mathbf{c}$,

- $\text{width}(\mathbf{c}, P) = \max\{\, \mathbf{c}^T\mathbf{x} \,|\, \mathbf{x} \in P \,\} - \min\{\, \mathbf{c}^T\mathbf{x} \,|\, \mathbf{x} \in P \,\} = \gamma - \delta.$

- **branching on $\mathbf{c}^T\mathbf{x}$** means creating the branches $\mathbf{c}^T\mathbf{x} = \lceil\delta\rceil$, $\mathbf{c}^T\mathbf{x} = \lceil\delta\rceil + 1, \ldots, \mathbf{cx} = \lfloor\gamma\rfloor$ (add constraint to LP relaxation).

- no branches created $\Rightarrow$ P has no integral point

Branching for IP feasibility

Given polyhedron P, integral vector $\mathbf{c}$,

- $\mathrm{width}(\mathbf{c}, P) = \max\{\, \mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P \,\} - \min\{\, \mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P \,\} = \gamma - \delta.$
- **branching on** $\mathbf{c}^T\mathbf{x}$ means creating the branches $\mathbf{c}^T\mathbf{x} = \lceil \delta \rceil$, $\mathbf{c}^T\mathbf{x} = \lceil \delta \rceil + 1, \ldots, \mathbf{c}\mathbf{x} = \lfloor \gamma \rfloor$ (add constraint to LP relaxation).
- no branches created $\Rightarrow$ P has no integral point
- when $\mathbf{c} = \mathbf{e}_i$, we are branching on single variable x_i

Branching for IP feasibility

Given polyhedron P, integral vector $\mathbf{c}$,

- $\text{width}(\mathbf{c}, P) = \max\{\, \mathbf{c}^T\mathbf{x} \,|\, \mathbf{x} \in P \,\} - \min\{\, \mathbf{c}^T\mathbf{x} \,|\, \mathbf{x} \in P \,\} = \gamma - \delta.$
- **branching on** $\mathbf{c}^T\mathbf{x}$ means creating the branches $\mathbf{c}^T\mathbf{x} = \lceil\delta\rceil$, $\mathbf{c}^T\mathbf{x} = \lceil\delta\rceil + 1, \ldots, \mathbf{c}\mathbf{x} = \lfloor\gamma\rfloor$ (add constraint to LP relaxation).
- no branches created $\Rightarrow$ P has no integral point
- when $\mathbf{c} = \mathbf{e}_i$, we are branching on single variable x_i
- different choices of $\mathbf{c}$ produce very different effects on branching

Column-BR in rangespace (CBR-R)

$$\begin{aligned} P &= \{\, \mathbf{x} \,|\, \boldsymbol{\ell} \leq \; A\mathbf{x} \leq \mathbf{b} \,\} \\ \tilde{P} &= \{\, \mathbf{y} \,|\, \boldsymbol{\ell} \leq (AU)\mathbf{y} \leq \mathbf{b}\} \end{aligned}$$

where U is unimodular

Column-BR in rangespace (CBR-R)

$$
\begin{aligned}
P &= \{\, \mathbf{x} \mid \boldsymbol{\ell} \leq A\mathbf{x} \leq \mathbf{b} \,\} \\
\tilde{P} &= \{\, \mathbf{y} \mid \boldsymbol{\ell} \leq (AU)\mathbf{y} \leq \mathbf{b} \}
\end{aligned}
$$

where U is unimodular

- There is 1-1 correspondence between $P \cap \mathbb{Z}^n$ and $\tilde{P} \cap \mathbb{Z}^n$ given by

$$
U\mathbf{y} = \mathbf{x}
$$

Column-BR in rangespace (CBR-R)

$$P = \{\, \mathbf{x} \,|\, \boldsymbol{\ell} \le A\mathbf{x} \le \mathbf{b} \,\}$$
$$\tilde{P} = \{\, \mathbf{y} \,|\, \boldsymbol{\ell} \le (AU)\mathbf{y} \le \mathbf{b}\}$$

where U is unimodular

- There is 1-1 correspondence between $P \cap \mathbb{Z}^n$ and $\tilde{P} \cap \mathbb{Z}^n$ given by

$$U\mathbf{y} = \mathbf{x}$$

- choose U s.t. columns of AU are reduced
- applies same even if some of the "$\le$" are "$=$"

Column-BR in rangespace (CBR-R)

$$
\begin{aligned}
P &= \{\mathbf{x} \mid \boldsymbol{\ell} \leq A\mathbf{x} \leq \mathbf{b}\} \\
\tilde{P} &= \{\mathbf{y} \mid \boldsymbol{\ell} \leq (AU)\mathbf{y} \leq \mathbf{b}\}
\end{aligned}
$$

where U is unimodular

- There is 1-1 correspondence between $P \cap \mathbb{Z}^n$ and $\tilde{P} \cap \mathbb{Z}^n$ given by

$$
U\mathbf{y} = \mathbf{x}
$$

- choose U s.t. columns of AU are reduced
- applies same even if some of the "$\leq$" are "$=$"
- a "primal" method

Column-BR in nullspace (CBR-N)

- Let $A_1\mathbf{x} = \mathbf{b}_1$ is a subset of the inequalities in $\boldsymbol{\ell} \leq A\mathbf{x} \leq \mathbf{b}$

Column-BR in nullspace (CBR-N)

- Let $A_1\mathbf{x} = \mathbf{b}_1$ is a subset of the inequalities in $\boldsymbol{\ell} \leq A\mathbf{x} \leq \mathbf{b}$
- reformulation using Hermite Normal Form (HNF) computation

$$\{\, \mathbf{x} \in \mathbb{Z}^n \,|\, A_1\mathbf{x} = \mathbf{b}_1 \,\} \;=\; \{\, \mathbf{x}_d + B_1\boldsymbol{\lambda} \,|\, \boldsymbol{\lambda} \in \mathbb{Z}^{n-m} \,\}$$

with $[B_1, \mathbf{x}_d]$ typically not reduced

Column-BR in nullspace (CBR-N)

- Let $A_1\mathbf{x} = \mathbf{b}_1$ is a subset of the inequalities in $\boldsymbol{\ell} \leq A\mathbf{x} \leq \mathbf{b}$
- reformulation using Hermite Normal Form (HNF) computation

$$\{\, \mathbf{x} \in \mathbb{Z}^n \,|\, A_1\mathbf{x} = \mathbf{b}_1 \,\} \;\; = \;\; \{\, \mathbf{x}_d + B_1\boldsymbol{\lambda} \,|\, \boldsymbol{\lambda} \in \mathbb{Z}^{n-m} \,\}$$

  with $[B_1, \mathbf{x}_d]$ typically not reduced

- substitute $B_1\boldsymbol{\lambda} + \mathbf{x}_d$ for $\mathbf{x}$, then do CBR-R

Column-BR in nullspace (CBR-N)

- Let $A_1\mathbf{x} = \mathbf{b}_1$ is a subset of the inequalities in $\boldsymbol{\ell} \leq A\mathbf{x} \leq \mathbf{b}$

- reformulation using Hermite Normal Form (HNF) computation

$$\{\, \mathbf{x} \in \mathbb{Z}^n \,|\, A_1\mathbf{x} = \mathbf{b}_1 \,\} \;\; = \;\; \{\, \mathbf{x}_d + B_1\boldsymbol{\lambda} \,|\, \boldsymbol{\lambda} \in \mathbb{Z}^{n-m} \,\}$$

  with $[B_1, \mathbf{x}_d]$ typically not reduced

- substitute $B_1\boldsymbol{\lambda} + \mathbf{x}_d$ for $\mathbf{x}$, then do CBR-R

- a "dual" method

CBR-N v/s CBR-R

- numerical output of CBR-N is similar to the reformulation technique of Aardal, Hurkens, and Lenstra (1998) – going from n vars, m equations to $n - m$ vars, no equations

CBR-N v/s CBR-R

- numerical output of CBR-N is similar to the reformulation technique of Aardal, Hurkens, and Lenstra (1998) – going from n vars, m equations to $n - m$ vars, no equations

- CBR-R stays in same space; computed in a simpler way

CBR-N v/s CBR-R

- numerical output of CBR-N is similar to the reformulation technique of Aardal, Hurkens, and Lenstra (1998) – going from n vars, m equations to $n - m$ vars, no equations

- CBR-R stays in same space; computed in a simpler way

- add slacks to "$\leq$", then apply AHL-reformulation?
  Was not tried. Going to nullspace has some benefits (esp. in cryptography applications)

CBR-N v/s CBR-R

- numerical output of CBR-N is similar to the reformulation technique of Aardal, Hurkens, and Lenstra (1998) – going from n vars, m equations to $n - m$ vars, no equations

- CBR-R stays in same space; computed in a simpler way

- add slacks to "$\leq$", then apply AHL-reformulation?
  Was not tried. Going to nullspace has some benefits (esp. in cryptography applications)

- both CBR-R and CBR-N actually work for essentially all hard IPs used to test "non-traditional" IP algorithms

$t+1$-level decomposable knapsack problems

- $\mathbf{a} = \mathbf{p}_1 M_1 + \mathbf{p}_2 M_2 + \cdots + \mathbf{p}_t M_t + \mathbf{r}$, with $M_1 > M_2 > \cdots > M_t$ and for suitable β, δ

$$(KP) \quad \beta \leq \mathbf{a}\,\mathbf{x} \;\leq \beta + \delta, \qquad \mathbf{0} \leq \mathbf{x} \leq \mathbf{u}, \quad \mathbf{x} \in \mathbb{Z}^n$$

$t+1$-level decomposable knapsack problems

- $\mathbf{a} = \mathbf{p}_1 M_1 + \mathbf{p}_2 M_2 + \cdots + \mathbf{p}_t M_t + \mathbf{r}$, with $M_1 > M_2 > \cdots > M_t$ and for suitable β, δ

$$(KP) \quad \beta \leq \mathbf{a}\,\mathbf{x} \leq \beta + \delta, \qquad \mathbf{0} \leq \mathbf{x} \leq \mathbf{u}, \quad \mathbf{x} \in \mathbb{Z}^n$$

- with suitably chosen data, the problem is "both hard and easy"

$t+1$-level decomposable knapsack problems

- $\mathbf{a} = \mathbf{p}_1 M_1 + \mathbf{p}_2 M_2 + \cdots + \mathbf{p}_t M_t + \mathbf{r}$, with $M_1 > M_2 > \cdots > M_t$ and for suitable β, δ

$$(KP) \quad \beta \leq \mathbf{a}\,\mathbf{x} \leq \beta + \delta, \qquad \mathbf{0} \leq \mathbf{x} \leq \mathbf{u}, \quad \mathbf{x} \in \mathbb{Z}^n$$

- with suitably chosen data, the problem is "both hard and easy"

- if $t = 1$, we just write $\mathbf{p} = \mathbf{p}_1$, $M = M_1$

DKP instances for $t = 1$

- a classic example: Jeroslow's problem

$$\begin{array}{rcl} 2(x_1 + \cdots + x_n) & = & n \\ x_i \in \{0, 1\}^n & & \end{array}$$

  where n is odd.

- with B&B branching on the x_i, no node is pruned above level $n/2$
- if we branch on $x_1 + \cdots + x_n$, we solve it at the root node
- here $\mathbf{p} = \mathbf{1}$, $\mathbf{r} = \mathbf{0}$, $M = 2$

Other known instances for $t = 1$

- $\mathbf{p} = \mathbf{1}$, $\mathbf{r} = (2^0, \ldots, 2^{n-1})$, $\mathbf{u} = \mathbf{1}$, $M = 2^{n+\ell+1}$: Todd's problem from Chvátal "Hard knapsack problems" (1980)

- $\mathbf{p} = \mathbf{1}$, $\mathbf{r} = (1, \ldots, n)$, $\mathbf{u} = \mathbf{1}$, $M = n(n+1)$: Avis' problem from same paper

- Gu, Nemhauser, Savelsbergh (2001) - modification of Todd's problem

- Cornuéjols, Urbaniak, Weismantel, Wolsey (1996): $\mathbf{p} > \mathbf{0}$, $\mathbf{u} = +\infty$ (inequality)

- Aardal-Lenstra (2004, 2006) : same as CUWW, but equality

Other known instances for $t = 1$

- $\mathbf{p} = \mathbf{1}$, $\mathbf{r} = (2^0, \ldots, 2^{n-1})$, $\mathbf{u} = \mathbf{1}$, $M = 2^{n+\ell+1}$: Todd's problem from Chvátal "Hard knapsack problems" (1980)

- $\mathbf{p} = \mathbf{1}$, $\mathbf{r} = (1, \ldots, n)$, $\mathbf{u} = \mathbf{1}$, $M = n(n+1)$: Avis' problem from same paper

- Gu, Nemhauser, Savelsbergh (2001) - modification of Todd's problem

- Cornuéjols, Urbaniak, Weismantel, Wolsey (1996): $\mathbf{p} > \mathbf{0}$, $\mathbf{u} = +\infty$ (inequality)

- Aardal-Lenstra (2004, 2006) : same as CUWW, but equality

All, except the last two take $\geq 2^{n/2}$ nodes for ordinary B&B.
In last, $\exists$ a large rhs for which the problem is infeasible

Recipe for DKPs and hardness

- Krishnamoorthy and Pataki (2009): unifying "recipe" to generate DKPs with $t = 1$

Recipe for DKPs and hardness

- Krishnamoorthy and Pataki (2009): unifying "recipe" to generate DKPs with $t = 1$
- Input: $\mathbf{p}, \mathbf{r}, \mathbf{u}$.

  Output: M, β, δ s.t. the infeasibility of (KP) is proven by branching on $\mathbf{p}^T\mathbf{x}$.

Recipe for DKPs and hardness

- Krishnamoorthy and Pataki (2009): unifying "recipe" to generate DKPs with $t = 1$

- Input: $\mathbf{p}, \mathbf{r}, \mathbf{u}$.

  Output: M, β, δ s.t. the infeasibility of (KP) is proven by branching on $\mathbf{p}^T\mathbf{x}$.

- lower bound on the number of nodes necessary for *ordinary* B&B (using x_j's)

DKPs get harder as t grows

two infeasible knapsack problems: can you tell which one is harder?

$$\begin{aligned} 1473x_1 + 1524x_2 + 1569x_3 + 1570x_4 + 1575x_5 + 1624x_6 + 1625x_7 \\ +2160x_8 + 2206x_9 + 2207x_{10} + 2211x_{11} + 2211x_{12} + 2257x_{13} \\ +2260x_{14} + 2305x_{15} + 2843x_{16} + 2943x_{17} + 2947x_{18} + 2991x_{19} \\ +2993x_{20} + 2997x_{21} + 3528x_{22} + 3577x_{23} + 3631x_{24} + 3677x_{25} \\ = 28980,\ x_i \in \{0, 1\}. \end{aligned}$$

$$\begin{aligned} 1314x_1 + 1315x_2 + 1317x_3 + 1318x_4 + 1971x_5 + 1972x_6 + 1973x_7 \\ +1976x_8 + 1977x_9 + 1977x_{10} + 2629x_{11} + 2630x_{12} + 2631x_{13} \\ +2631x_{14} + 2633x_{15} + 2634x_{16} + 2635x_{17} + 2635x_{18} + 3287x_{19} \\ +3287x_{20} + 3287x_{21} + 3289x_{22} + 3292x_{23} + 3293x_{24} + 3293x_{25} \\ = 28981,\ x_i \in \{0, 1\}. \end{aligned}$$

Similar looking DKPs

- The second one has $t = 1$, and takes $\approx 22,000$ nodes to prove infeasibility.

- The first one has $t = 2$, and takes ≈ 3.6 million nodes to prove infeasibility. (Note that $2^{25} \approx 33\,\mathrm{million}$).

DKPs get more interesting as t grows

- if $Q =$ LP relaxation of $t = 2$ DKP, then $\mathrm{width}(\mathbf{e}_i, Q) = 1 - 0 \; \forall i$

DKPs get more interesting as t grows

- if $Q =$ LP relaxation of $t = 2$ DKP, then $\mathrm{width}(\mathbf{e}_i, Q) = 1 - 0 \ \forall i$

- $\mathrm{width}(\mathbf{p}_1, Q) > 1,$ but $[\max\{\mathbf{p}_1\mathbf{x} : \mathbf{x} \in Q\} - \min\{\mathbf{p}_1\mathbf{x} : \mathbf{x} \in Q\}]$ only contains one integer. So "branching" on $\mathbf{p}_1\mathbf{x}$ means adding $\mathbf{p}_1\mathbf{x} = k$ to the LP for some k

DKPs get more interesting as t grows

- if Q = LP relaxation of $t = 2$ DKP, then $\mathrm{width}(\mathbf{e}_i, Q) = 1 - 0 \; \forall i$

- $\mathrm{width}(\mathbf{p}_1, Q) > 1$, but $[\max\{\mathbf{p}_1\mathbf{x} : \mathbf{x} \in Q\} - \min\{\mathbf{p}_1\mathbf{x} : \mathbf{x} \in Q\}]$ only contains one integer. So "branching" on $\mathbf{p}_1\mathbf{x}$ means adding $\mathbf{p}_1\mathbf{x} = k$ to the LP for some k

- afterwards, branching on $\mathbf{p}_2\mathbf{x}$ proves infeasibility

DKPs get more interesting as t grows

- if Q = LP relaxation of $t = 2$ DKP, then $\mathrm{width}(\mathbf{e}_i, Q) = 1 - 0\ \forall i$

- $\mathrm{width}(\mathbf{p}_1, Q) > 1$, but $[\max\{\mathbf{p}_1\mathbf{x} : \mathbf{x} \in Q\} - \min\{\mathbf{p}_1\mathbf{x} : \mathbf{x} \in Q\}]$ only contains one integer. So "branching" on $\mathbf{p}_1\mathbf{x}$ means adding $\mathbf{p}_1\mathbf{x} = k$ to the LP for some k

- afterwards, branching on $\mathbf{p}_2\mathbf{x}$ proves infeasibility

- These DKPs are called *cascade* problems. For $n = 40$ they become unsolvable as IPs for commercial solvers

DKPs get more interesting as t grows

- if $Q =$ LP relaxation of $t = 2$ DKP, then $\mathrm{width}(\mathbf{e}_i, Q) = 1 - 0\ \forall i$

- $\mathrm{width}(\mathbf{p}_1, Q) > 1,$ but $[\max\{\mathbf{p}_1\mathbf{x} : \mathbf{x} \in Q\} - \min\{\mathbf{p}_1\mathbf{x} : \mathbf{x} \in Q\}]$ only contains one integer. So "branching" on $\mathbf{p}_1\mathbf{x}$ means adding $\mathbf{p}_1\mathbf{x} = k$ to the LP for some k

- afterwards, branching on $\mathbf{p}_2\mathbf{x}$ proves infeasibility

- These DKPs are called *cascade* problems. For $n = 40$ they become unsolvable as IPs for commercial solvers

- a "not thin" direction beats a thin direction!

Easiness of DKPs

- they are easy, if branching on $\mathbf{p}_1^T\mathbf{x}$, $\mathbf{p}_2^T\mathbf{x}$, . . . , $\mathbf{p}_t^T\mathbf{x}$.

Easiness of DKPs

- they are easy, if branching on $\mathbf{p}_1^T\mathbf{x}$, $\mathbf{p}_2^T\mathbf{x}$, . . . , $\mathbf{p}_t^T\mathbf{x}$.
- if we fix $\mathbf{p}_i^T\mathbf{x},$ the problem simplifies ($\mathbf{p}_i M_i$ disappears)

Easiness of DKPs

- they are easy, if branching on $\mathbf{p}_1^T\mathbf{x}$, $\mathbf{p}_2^T\mathbf{x}$, . . . , $\mathbf{p}_t^T\mathbf{x}$.
- if we fix $\mathbf{p}_i^T\mathbf{x}$, the problem simplifies ($\mathbf{p}_i M_i$ disappears)
- width in direction of $\mathbf{p}_{s+1}\mathbf{x}$, after we branched on $\mathbf{p}_1\mathbf{x}, \ldots, \mathbf{p}_s\mathbf{x}$ is

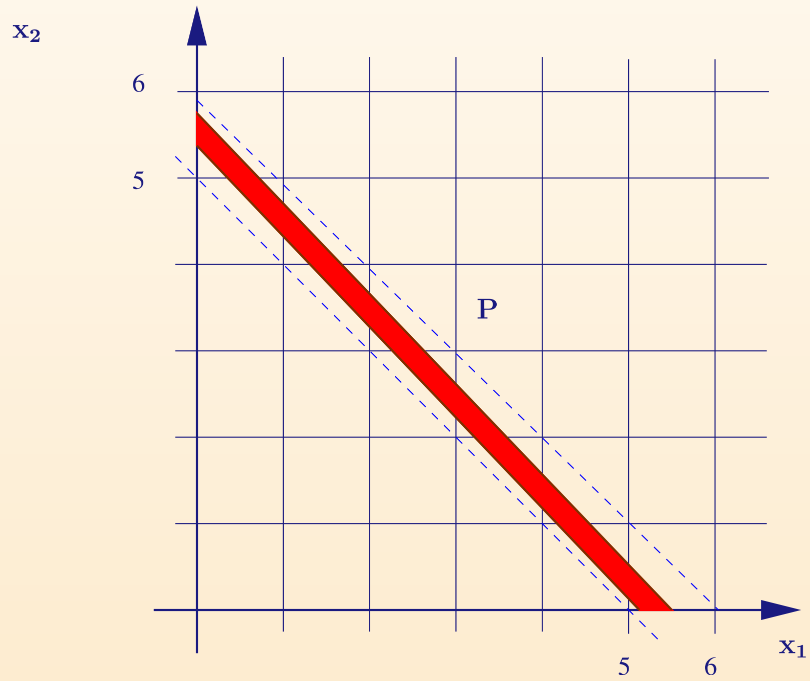
$$O\left(\frac{rhs}{M_{s+1}^2} + \frac{\delta}{M_{s+1}}\right).$$

CBR's action on DKPs

- Briefly: the "good reasons" for $\mathbf{p}_i\mathbf{x}$ are transferred to the variable y_{n-i} in the reformulation

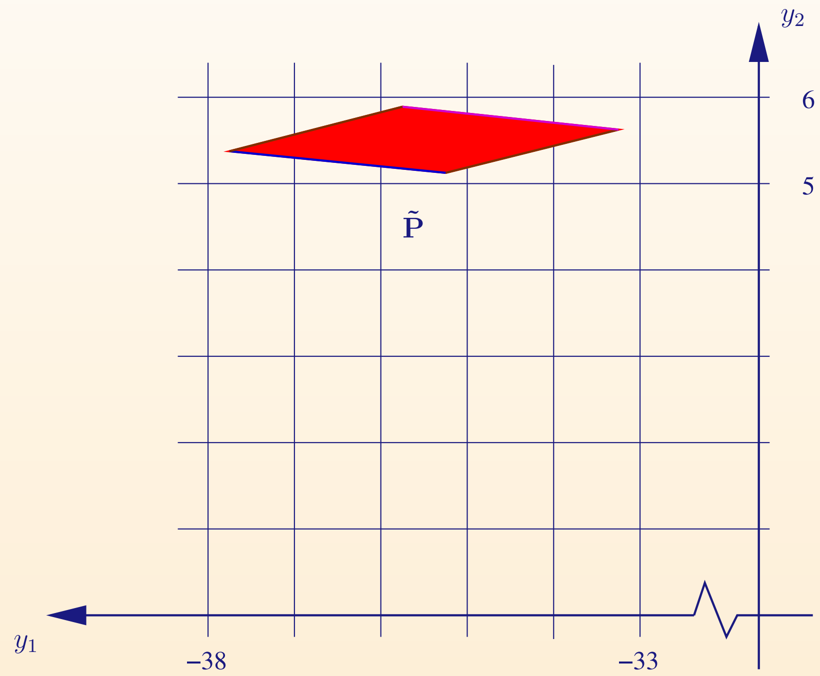
Example of CBR of a DKP

$$\begin{array}{rcl} 106 \leq & 21x_1 + 19x_2 & \leq 113 \\ x_1, x_2 \in & & \in [0,6] \cap \mathbb{Z} \end{array}$$



Hard for branching on x_is. Easy for branching on $x_1 + x_2$: $\max = 5.94$, $\min = 5.04$.

After Reformulation ...



. . . branching on y_2 proves infeasibility!

CBR's Action on DKP

- we compute U so that

$$\begin{pmatrix} \sum_{i=1}^{t} \mathbf{p}_i M_i + \mathbf{r} \\ I \end{pmatrix} U \quad \text{is reduced.}$$

CBR's Action on DKP

- we compute U so that

$$\begin{pmatrix} \sum_{i=1}^{t} \mathbf{p}_i M_i + \mathbf{r} \\ I \end{pmatrix} U \quad \text{is reduced.}$$

- **Theorem:** If separation between $M_1 > M_2 > \cdots > M_t$ is suitably large, then

$$\begin{pmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_t \end{pmatrix} U = \begin{pmatrix} 0 & 0 \ldots & 0 & 0 & 0 & * \\ 0 & 0 \ldots & 0 & 0 & * & * \\ \vdots & & & & & \\ 0 & 0 \ldots & * & \ldots & * & * \end{pmatrix}$$

CBR's Action on DKP

- we compute U so that

$$\begin{pmatrix} \sum_{i=1}^{t} \mathbf{p}_i M_i + \mathbf{r} \\ I \end{pmatrix} U \quad \text{is reduced.}$$

- **Theorem:** If separation between $M_1 > M_2 > \cdots > M_t$ is suitably large, then

$$\begin{pmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_t \end{pmatrix} U = \begin{pmatrix} 0 & 0 \ldots & 0 & 0 & 0 & * \\ 0 & 0 \ldots & 0 & 0 & * & * \\ \vdots & & & & & \\ 0 & 0 \ldots & * & \ldots & * & * \end{pmatrix}$$

When computing U, we do not know the decomposition!

CBR's Action on DKP

- using the correspondence $U\mathbf{y} = \mathbf{x}$, we get

$$\mathbf{p}_1\mathbf{x} = \mathbf{p}_1(U\mathbf{y}) = (\mathbf{p}_1 U)\mathbf{y} = (\mathbf{p}_1 U)_n y_n.$$

- **Corollary:**
  - Branching on y_n in reformulation $\Leftrightarrow$ branching on $\mathbf{p}_1\mathbf{x}$ in original problem
  - Afterwards: $y_{n-1} \Leftrightarrow \mathbf{p}_2\mathbf{x}$, etc.
  - Analogous result for CBR-N

Summary of CBR

- general reformulation technique for arbitrary IPs.
- has two variants: CBR-R and CBR-N, both work in practice and can be analyzed
- a fairly general class of IPs provably hard for ordinary B&B
- the provably hard problems turn into provably easy ones: the reformulation "uncovers" the hidden, dominant directions
- The *cascade* problems: thinner $\neq$ better!

Summary of CBR

- general reformulation technique for arbitrary IPs.
- has two variants: CBR-R and CBR-N, both work in practice and can be analyzed
- a fairly general class of IPs provably hard for ordinary B&B
- the provably hard problems turn into provably easy ones: the reformulation "uncovers" the hidden, dominant directions
- The *cascade* problems: thinner $\neq$ better!
- Pataki et al. (2010) - B&B solves "almost all" instances of CBR-R of $\{\mathbf{x} \mid \boldsymbol{\ell}_1 \leq A\mathbf{x} \leq \mathbf{u}_1;\ \boldsymbol{\ell}_2 \leq \mathbf{x} \leq \mathbf{u}_2\}$ at root node if $A_{ij} \in U\{1, \ldots, M\}$ for ***sufficiently large*** M

Number Partitioning Problem (NPP)

- Given $S = \{a_1, \ldots, a_n\}$ with $a_j \in \mathbb{Z}_{>0}$, $\alpha = \sum_j a_j$,

Number Partitioning Problem (NPP)

- Given $S = \{a_1, \ldots, a_n\}$ with $a_j \in \mathbb{Z}_{>0}$, $\alpha = \sum_j a_j$,
- divide S into 2 disjoint subsets $S_1 \cup S_2 = S$ such that

Number Partitioning Problem (NPP)

- Given $S = \{a_1, \ldots, a_n\}$ with $a_j \in \mathbb{Z}_{>0}$, $\alpha = \sum_j a_j$,
- divide S into 2 disjoint subsets $S_1 \cup S_2 = S$ such that

$$\triangle = \left| \sum_{j \in S_1} a_j - \sum_{j \in S_2} a_j \right|,$$

Number Partitioning Problem (NPP)

- Given $S = \{a_1, \ldots, a_n\}$ with $a_j \in \mathbb{Z}_{>0}$, $\alpha = \sum_j a_j$,
- divide S into 2 disjoint subsets $S_1 \cup S_2 = S$ such that

$$\triangle = \left| \sum_{j \in S_1} a_j \; - \; \sum_{j \in S_2} a_j \right|,$$ the *discrepancy*,

Number Partitioning Problem (NPP)

- Given $S = \{a_1, \ldots, a_n\}$ with $a_j \in \mathbb{Z}_{>0}$, $\alpha = \sum_j a_j$,
- divide S into 2 disjoint subsets $S_1 \cup S_2 = S$ such that

$$\triangle = \left| \sum_{j \in S_1} a_j \; - \; \sum_{j \in S_2} a_j \right|, \text{ the \textit{discrepancy},}$$

is minimized.

Number Partitioning Problem (NPP)

- Given $S = \{a_1, \ldots, a_n\}$ with $a_j \in \mathbb{Z}_{>0}$, $\alpha = \sum_j a_j$,
- divide S into 2 disjoint subsets $S_1 \cup S_2 = S$ such that

$$\triangle = \left| \sum_{j \in S_1} a_j \; - \; \sum_{j \in S_2} a_j \right|, \text{ the } \textit{discrepancy},$$

  is minimized.
- $\triangle^* =$ minimum discrepancy

Number Partitioning Problem (NPP)

- Given $S = \{a_1, \ldots, a_n\}$ with $a_j \in \mathbb{Z}_{>0}$, $\alpha = \sum_j a_j$,
- divide S into 2 disjoint subsets $S_1 \cup S_2 = S$ such that

$$\triangle = \left| \sum_{j \in S_1} a_j \; - \; \sum_{j \in S_2} a_j \right|,$$ the *discrepancy*,

  is minimized.

- $\triangle^* =$ minimum discrepancy
- allocate $\beta = 1/2\ \alpha$, or, as close as possible to β, to each subset

Number Partitioning Problem (NPP)

Number Partitioning Problem (NPP)

- one of six basic NP-complete problems in Garey and Johnson (79)

Number Partitioning Problem (NPP)

- one of six basic NP-complete problems in Garey and Johnson (79)
- only one dealing directly with *numbers*

Number Partitioning Problem (NPP)

- one of six basic NP-complete problems in Garey and Johnson (79)
- only one dealing directly with *numbers*
- *balanced* NPP ($\textsc{BalNPP}$): $|S_1| = |S_2| = n/2$ (for even n)

NPP – Example

- $S = \{6, 4, 7, 8, 5\}$

NPP – Example

- $S = \{\, 6, 4, 7, 8, 5 \,\}$

- $S = \{4, 5, 6, 7, 8\}$

NPP – Example

- $S = \{6, 4, 7, 8, 5\}$
- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 7\}$ with subset sum $= 16$,

NPP – Example

- $S = \{ 6, 4, 7, 8, 5 \}$
- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 7\}$ with subset sum $= 16$,
  - $S_2 = \{6, 8\}$ with subset sum $= 14$;

NPP – Example

- $S = \{\,6, 4, 7, 8, 5\,\}$

- $S = \{4, 5, 6, 7, 8\}$

  - $S_1 = \{4, 5, 7\}$ with subset sum $= 16$,
  - $S_2 = \{6, 8\}$ with subset sum $= 14$; $\quad \triangle = 2$

NPP – Example

- $S = \{\, 6, 4, 7, 8, 5 \,\}$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 7\}$ with subset sum $= 16$,
  - $S_2 = \{6, 8\}$ with subset sum $= 14$; $\quad \triangle = 2$

- $S = \{4, 5, 6, 7, 8\}$

NPP – Example

- $S = \{ 6, 4, 7, 8, 5 \}$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 7\}$ with subset sum $= 16$,
  - $S_2 = \{6, 8\}$ with subset sum $= 14$; $\quad \triangle = 2$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 6\}$ and $S_2 = \{7, 8\}$

NPP – Example

- $S = \{\,6, 4, 7, 8, 5\,\}$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 7\}$ with subset sum $= 16$,
  - $S_2 = \{6, 8\}$ with subset sum $= 14$; $\triangle = 2$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 6\}$ and $S_2 = \{7, 8\}$
  - both subset sums $= 15$;

NPP – Example

- $S = \{6, 4, 7, 8, 5\}$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 7\}$ with subset sum $= 16$,
  - $S_2 = \{6, 8\}$ with subset sum $= 14$; $\quad \triangle = 2$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 6\}$ and $S_2 = \{7, 8\}$
  - both subset sums $= 15$; $\quad \triangle = \triangle^* = 0$

NPP – Example

- $S = \{6, 4, 7, 8, 5\}$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 7\}$ with subset sum $= 16$,
  - $S_2 = \{6, 8\}$ with subset sum $= 14$; $\quad \triangle = 2$

- $S = \{4, 5, 6, 7, 8\}$
  - $S_1 = \{4, 5, 6\}$ and $S_2 = \{7, 8\}$
  - both subset sums $= 15$; $\quad \triangle = \triangle^* = 0$

- $\triangle^* = 0$ (or $\triangle^* = 1$ when α odd) gives a *perfect* partition

Applications of NPP

Applications of NPP

- practical

- theoretical

Applications of NPP

- practical
  - scheduling jobs on processors
    (NPP into $k \geq 3$ subsets: multiprocessor scheduling problem)
  - VLSI circuit design
  - public key cryptography
- theoretical

Applications of NPP

- practical
  - scheduling jobs on processors
    (NPP into $k \geq 3$ subsets: multiprocessor scheduling problem)
  - VLSI circuit design
  - public key cryptography
- theoretical
  - phase transition (fully characterized mathematically)
  - NP-completeness of other problems involving numbers –
    bin packing, knapsack etc.

NPP – known results

- $a_j = U[1, R]$ for $R \in \mathbb{Z}_{>0}$

NPP – known results

- $a_j = U[1, R]$ for $R \in \mathbb{Z}_{>0}$

- median and expected $\triangle^*$ (in the limit)

NPP – known results

- $a_j = U[1, R]$ for $R \in \mathbb{Z}_{>0}$
- median and expected $\triangle^*$ (in the limit)
  - $\triangle^* = O(\sqrt{n}\, 2^{-n}\, R)$ for NPP

NPP – known results

- $a_j = U[1, R]$ for $R \in \mathbb{Z}_{>0}$
- median and expected $\triangle^*$ (in the limit)
  - $\triangle^* = O(\sqrt{n}\, 2^{-n}\, R)$ for NPP
  - $\triangle^* = O(\ \ n\ 2^{-n}\, R)$ for BalNPP

NPP – known results

- $a_j = U[1, R]$ for $R \in \mathbb{Z}_{>0}$

- median and expected $\triangle^*$ (in the limit)

  – $\triangle^* = O(\sqrt{n}\, 2^{-n}\, R)$ for NPP
  – $\triangle^* = O(\;\; n\; 2^{-n}\, R)$ for BalNPP

    ∗ Karmarkar, Karp, Lueker, Odlyzko (88): median $\triangle^*$ for NPP
    ∗ Lueker (98): average $\triangle^*$ for NPP
    ∗ Mertens (98): median and average $\triangle^*$ for BalNPP

Phase transition of NPP and BalNPP

Phase transition of NPP and BalNPP

- Prob$(\triangle^* = 0/1) \to 1$ as $n \to \infty$ for $R < 2^n$

Phase transition of NPP and BalNPP

- Prob$(\triangle^* = 0/1) \to 1$ as $n \to \infty$ for $R < 2^n$ (easy phase)

Phase transition of NPP and BalNPP

- Prob$(\triangle^* = 0/1) \rightarrow 1$ as $n \rightarrow \infty$ for $R < 2^n$ (easy phase)
- Prob$(\triangle^* = 0/1) \rightarrow 0$ as $n \rightarrow \infty$ for $R > 2^n$

Phase transition of NPP and BalNPP

- Prob$(\triangle^* = 0/1) \to 1$ as $n \to \infty$ for $R < 2^n$ (easy phase)
- Prob$(\triangle^* = 0/1) \to 0$ as $n \to \infty$ for $R > 2^n$ (hard phase)

Phase transition of NPP and BalNPP

- Prob$(\triangle^* = 0/1) \to 1$ as $n \to \infty$ for $R < 2^n$ (easy phase)
- Prob$(\triangle^* = 0/1) \to 0$ as $n \to \infty$ for $R > 2^n$ (hard phase)
  - Gent and Walsh (96): empirical evidence
  - Mertens (98): spin glass analogy
  - Borgs, Chayes, and Pittel (01): complete mathematical analysis

Phase transition of NPP and BalNPP

- Prob($\triangle^* = 0/1) \rightarrow 1$ as $n \rightarrow \infty$ for $R < 2^n$ (easy phase)
- Prob($\triangle^* = 0/1) \rightarrow 0$ as $n \rightarrow \infty$ for $R > 2^n$ (hard phase)
  - Gent and Walsh (96): empirical evidence
  - Mertens (98): spin glass analogy
  - Borgs, Chayes, and Pittel (01): complete mathematical analysis
- \# perfect partitions $\uparrow$ as $R \downarrow$ with $R < 2^n$

Phase transition of NPP and BalNPP

- Prob$(\triangle^* = 0/1) \to 1$ as $n \to \infty$ for $R < 2^n$ (easy phase)
- Prob$(\triangle^* = 0/1) \to 0$ as $n \to \infty$ for $R > 2^n$ (hard phase)
  - Gent and Walsh (96): empirical evidence
  - Mertens (98): spin glass analogy
  - Borgs, Chayes, and Pittel (01): complete mathematical analysis
- $\#$ perfect partitions $\uparrow$ as $R \downarrow$ with $R < 2^n$
- minimum partition unique for $R \gg 2^n$

Karmarkar-Karp differencing (KK)

Karmarkar-Karp differencing (KK)

- maintain sorted list of numbers

Karmarkar-Karp differencing (KK)

- maintain sorted list of numbers
- replace two largest numbers by their difference
  (commit to place them in opposite subsets)

Karmarkar-Karp differencing (KK)

- maintain sorted list of numbers
- replace two largest numbers by their difference (commit to place them in opposite subsets)
- Yakir (96): $\triangle_{KK} = O(n^{-0.72 \log n} R)$

Karmarkar-Karp differencing (KK)

- maintain sorted list of numbers
- replace two largest numbers by their difference
  (commit to place them in opposite subsets)
- Yakir (96): $\triangle_{KK} = O(n^{-0.72 \log n} R)$
  recall, $\triangle^* = O(\sqrt{n}\, 2^{-n} R)$

Karmarkar-Karp differencing (KK)

- maintain sorted list of numbers
- replace two largest numbers by their difference
  (commit to place them in opposite subsets)
- Yakir (96): $\triangle_{KK} = O(n^{-0.72 \log n} R)$
  recall, $\triangle^* = O(\sqrt{n}\, 2^{-n} R)$
- running time is $O(n \log n)$

Complete KK heuristic

Complete KK heuristic

- Korf (98), Mertens (99)

Complete KK heuristic

- Korf (98), Mertens (99)
- also consider replacing two largest numbers by their *sum*

Complete KK heuristic

- Korf (98), Mertens (99)
- also consider replacing two largest numbers by their *sum*
- improves on KK discrepancy as it continues to run

Complete KK heuristic

- Korf (98), Mertens (99)
- also consider replacing two largest numbers by their *sum*
- improves on KK discrepancy as it continues to run
- effective in practice in the easy phase

Complete KK heuristic

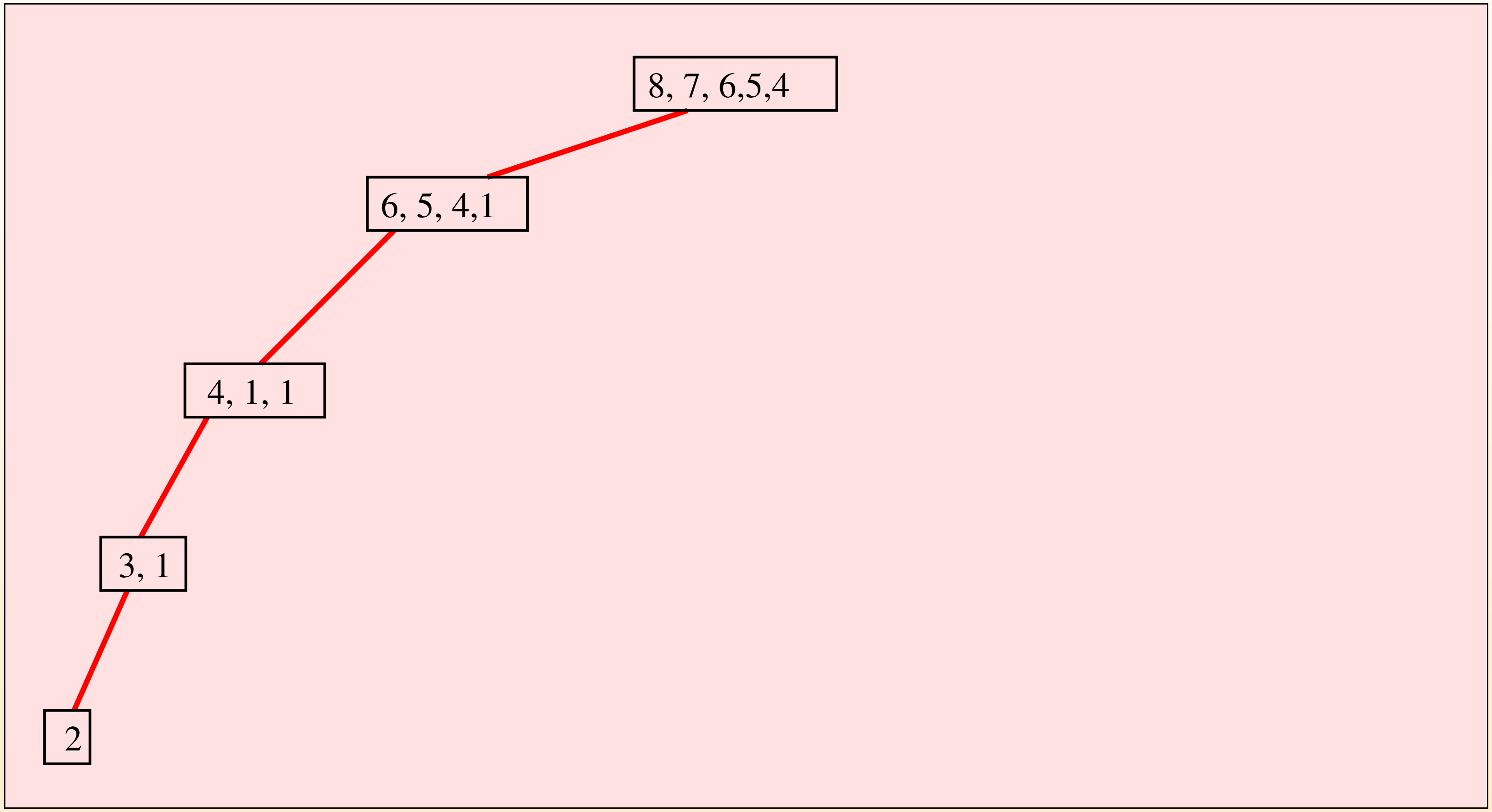
- Korf (98), Mertens (99)
- also consider replacing two largest numbers by their *sum*
- improves on KK discrepancy as it continues to run
- effective in practice in the easy phase
- $\#$ of branch-and-bound nodes is exponential in n when $R > 2^n$

Complete KK heuristic

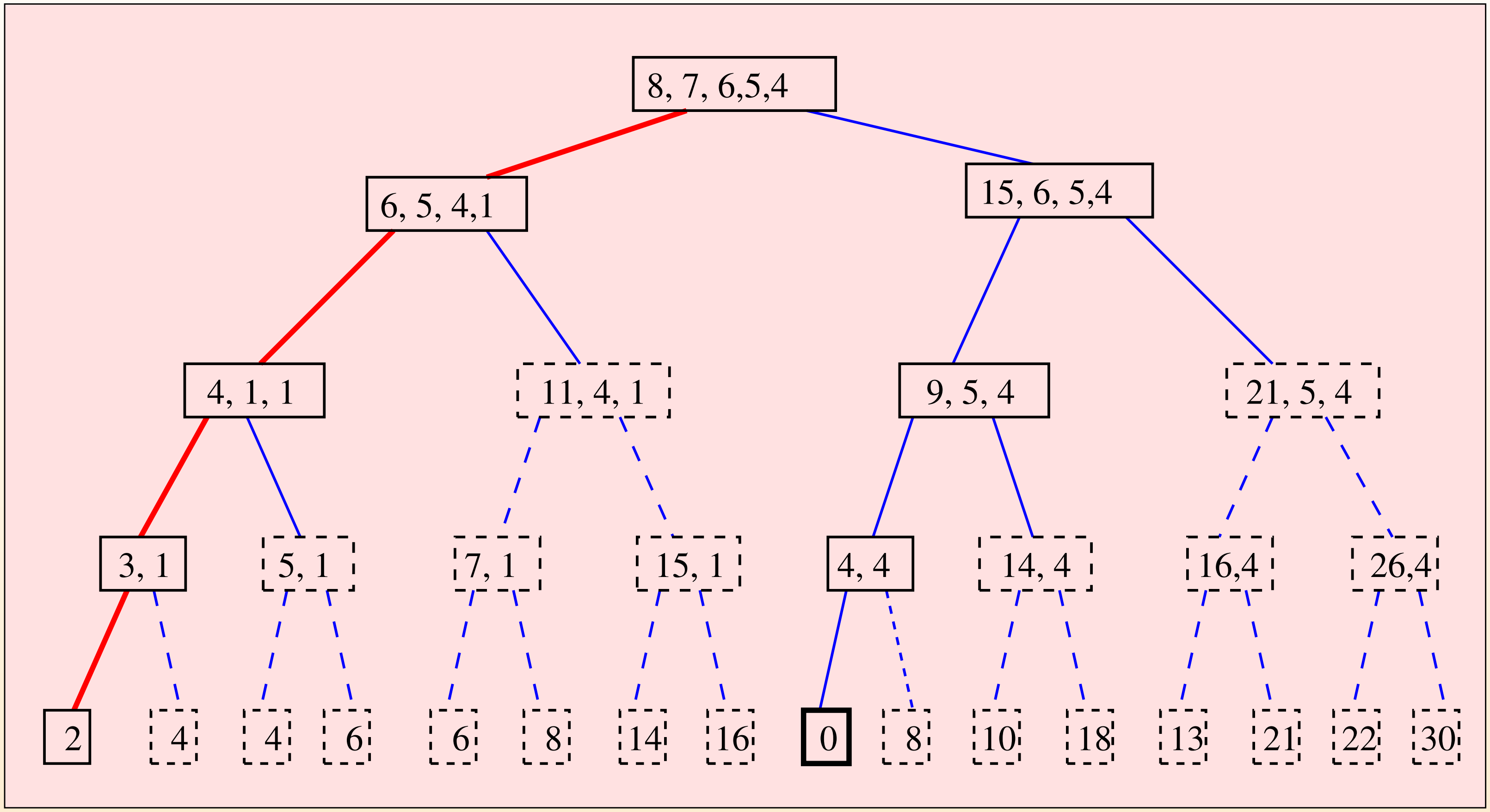
- Korf (98), Mertens (99)
- also consider replacing two largest numbers by their *sum*
- improves on KK discrepancy as it continues to run
- effective in practice in the easy phase
- $\#$ of branch-and-bound nodes is exponential in n when $R > 2^n$
- converges very slowly

KK and CKK: Example

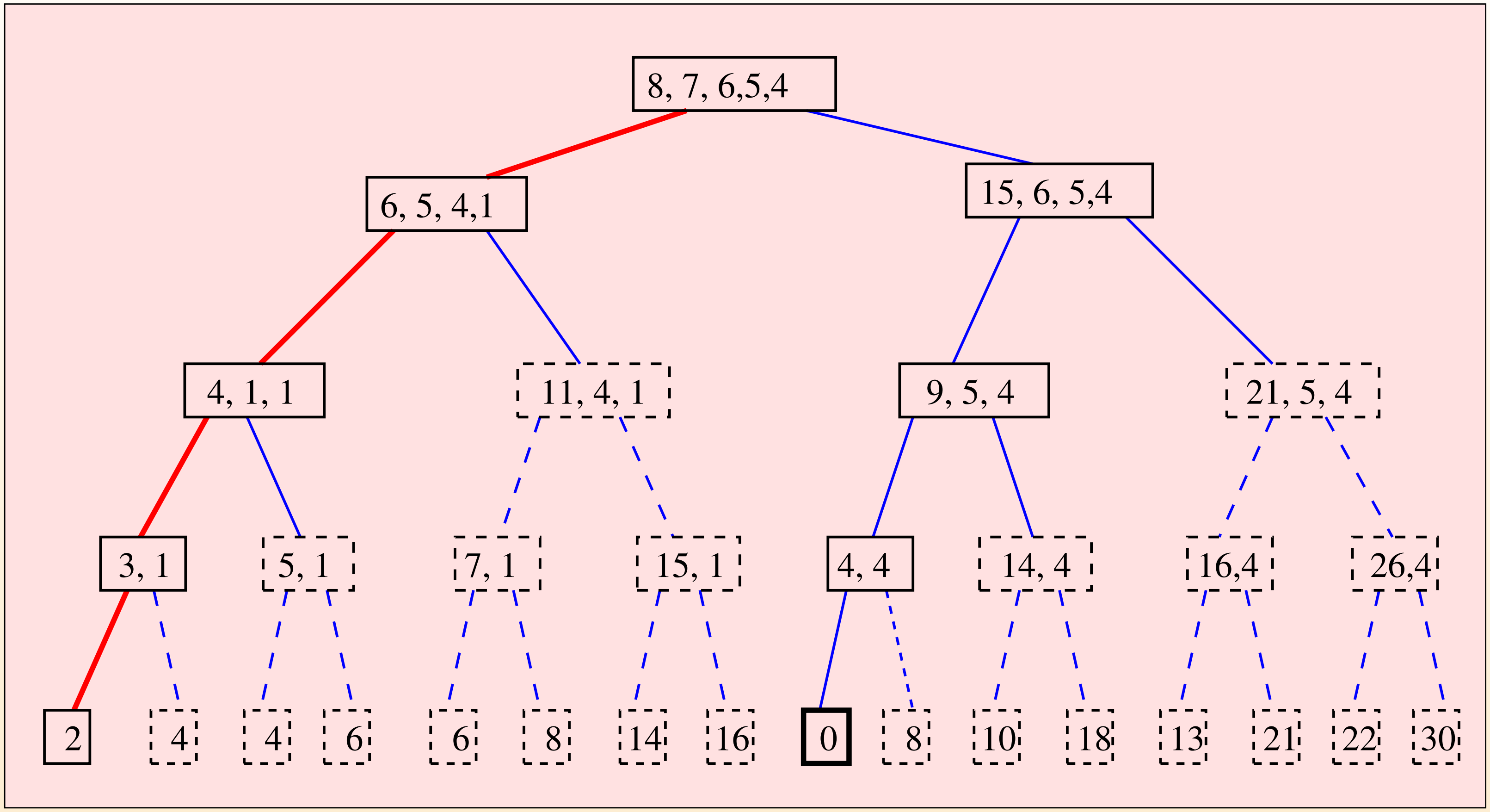
KK and CKK: Example



KK and CKK: Example

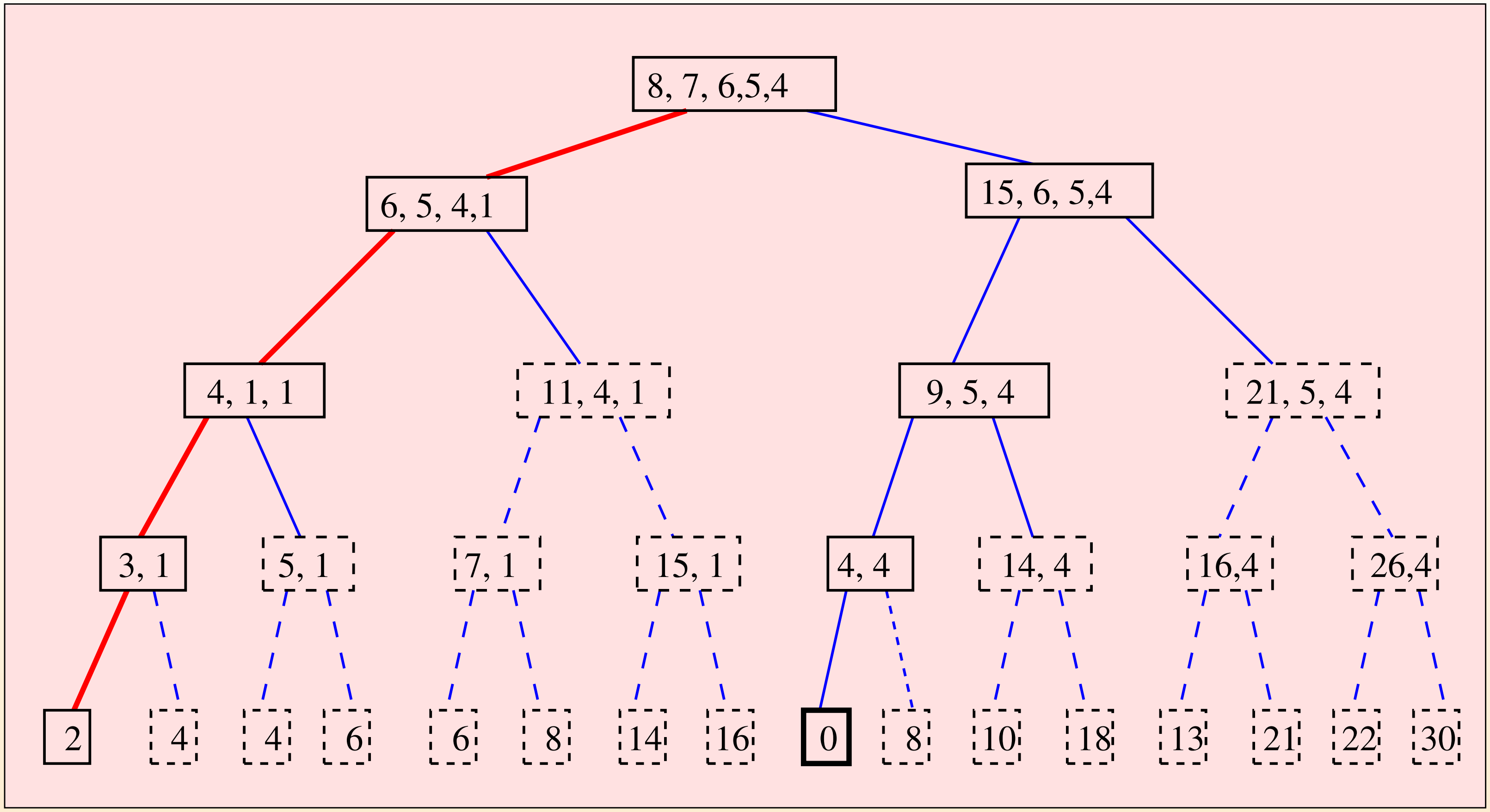


KK and CKK: Example



- dashed parts of the tree are pruned

KK and CKK: Example



- dashed parts of the tree are pruned
- two-color associated tree to recover partition

Algorithms for NPP

Algorithms for NPP

- KK is the *best* polynomial time approx. algo known

Algorithms for NPP

- KK is the *best* polynomial time approx. algo known
- metaheuristics for easy phase (Storer (96))

Algorithms for NPP

- KK is the *best* polynomial time approx. algo known
- metaheuristics for easy phase (Storer (96))
- concentrate on "hard phase" ($R > 2^n$)

Algorithms for NPP

- KK is the *best* polynomial time approx. algo known
- metaheuristics for easy phase (Storer (96))
- concentrate on "hard phase" ($R > 2^n$)
- lattice-based techniques?

Algorithms for NPP

- KK is the *best* polynomial time approx. algo known
- metaheuristics for easy phase (Storer (96))
- concentrate on "hard phase" $(R > 2^n)$
- lattice-based techniques?
- typical numbers are *huge*; for $n = 30$, look at a_j's with 11 digits!

Lattice Problems and NPP

Lattice Problems and NPP

- Closest Vector Problem (decision version - DCVP):

Lattice Problems and NPP

- Closest Vector Problem (decision version - DCVP):
  Given: lattice basis $B \in \mathbb{Z}^{m \times n}$, target vector $\mathbf{t}$, rational $\gamma > 0$,

Lattice Problems and NPP

- Closest Vector Problem (decision version - DCVP):
  Given: lattice basis $B \in \mathbb{Z}^{m\times n}$, target vector $\mathbf{t}$, rational $\gamma > 0$,
  find $\mathbf{x} \in \mathbb{Z}^n$ s.t. $\| B\mathbf{x} - \mathbf{t} \| \leq \gamma$, or prove $\| B\mathbf{x} - \mathbf{t} \| > \gamma \, \forall \, \mathbf{x} \in \mathbb{Z}^n$.

Lattice Problems and NPP

- Closest Vector Problem (decision version - DCVP):
  Given: lattice basis $B \in \mathbb{Z}^{m \times n}$, target vector $\mathbf{t}$, rational $\gamma > 0$, find $\mathbf{x} \in \mathbb{Z}^n$ s.t. $\| B\mathbf{x} - \mathbf{t} \| \leq \gamma$, or prove $\| B\mathbf{x} - \mathbf{t} \| > \gamma \, \forall \, \mathbf{x} \in \mathbb{Z}^n$.

- Decision version of NPP (DNPP_d): Given numbers $a_1, \ldots, a_n$ and an even number $2d$, decide if a partition exists with $\triangle \leq 2d$.

Lattice Problems and NPP

- Closest Vector Problem (decision version - DCVP):
  Given: lattice basis $B \in \mathbb{Z}^{m \times n}$, target vector $\mathbf{t}$, rational $\gamma > 0$, find $\mathbf{x} \in \mathbb{Z}^n$ s.t. $\| B\mathbf{x} - \mathbf{t} \| \leq \gamma$, or prove $\| B\mathbf{x} - \mathbf{t} \| > \gamma \, \forall \, \mathbf{x} \in \mathbb{Z}^n$.

- Decision version of NPP (DNPP_d): Given numbers $a_1, \ldots, a_n$ and an even number $2d$, decide if a partition exists with $\triangle \leq 2d$. Equivalently, find $\mathbf{x} \in \{0, 1\}^n$ s.t. $\sum_j a_j x_j = \beta - \delta$ for some $\delta \leq d$, if it exists.

Lattice Problems and NPP

- Closest Vector Problem (decision version - DCVP):
  Given: lattice basis $B \in \mathbb{Z}^{m \times n}$, target vector $\mathbf{t}$, rational $\gamma > 0$, find $\mathbf{x} \in \mathbb{Z}^n$ s.t. $\| B\mathbf{x} - \mathbf{t} \| \leq \gamma$, or prove $\| B\mathbf{x} - \mathbf{t} \| > \gamma \, \forall \, \mathbf{x} \in \mathbb{Z}^n$.

- Decision version of NPP (DNPP_d): Given numbers $a_1, \ldots, a_n$ and an even number $2d$, decide if a partition exists with $\triangle \leq 2d$. Equivalently, find $\mathbf{x} \in \{0,1\}^n$ s.t. $\sum_j a_j x_j = \beta - \delta$ for some $\delta \leq d$, if it exists. Here, $\beta = \sum_j a_j/2$.

Lattice Problems and NPP

- Closest Vector Problem (decision version - DCVP):
  Given: lattice basis $B \in \mathbb{Z}^{m \times n}$, target vector $\mathbf{t}$, rational $\gamma > 0$,
  find $\mathbf{x} \in \mathbb{Z}^n$ s.t. $\| B\mathbf{x} - \mathbf{t} \| \leq \gamma$, or prove $\| B\mathbf{x} - \mathbf{t} \| > \gamma \, \forall \, \mathbf{x} \in \mathbb{Z}^n$.

- Decision version of NPP (DNPP_d): Given numbers $a_1, \ldots, a_n$ and an even number $2d$, decide if a partition exists with $\triangle \leq 2d$. Equivalently, find $\mathbf{x} \in \{0,1\}^n$ s.t. $\sum_j a_j x_j = \beta - \delta$ for some $\delta \leq d$, if it exists. Here, $\beta = \sum_j a_j/2$.

- reduce DNPP to DCVP

DNPP **to** DCVP

Theorem 1. DNPP_d *is reducible to* DCVP *for* $d > 0$.

DNPP to DCVP

Theorem 1. DNPP_d *is reducible to* DCVP *for* $d > 0$.

$$B = \begin{bmatrix} 2d\, I \\ \mathbf{a}^T \end{bmatrix},$$

DNPP to DCVP

Theorem 1. DNPP_d *is reducible to* DCVP *for* $d > 0$.

$$B = \begin{bmatrix} 2d\, I \\ \mathbf{a}^T \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} d\, \mathbf{1} \\ \beta \end{bmatrix}.$$

DNPP **to** DCVP

Theorem 1. DNPP_d *is reducible to* DCVP *for* $d > 0$.

$$B = \begin{bmatrix} 2d\, I \\ \mathbf{a}^T \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} d\,\mathbf{1} \\ \beta \end{bmatrix}.$$

- output of reduction: DCVP instance $(B, \mathbf{t}, d\sqrt{n+1})$

DNPP **to** DCVP

Theorem 1. DNPP_d *is reducible to* DCVP *for* $d > 0$.

$$B = \begin{bmatrix} 2d\, I \\ \mathbf{a}^T \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} d\, \mathbf{1} \\ \beta \end{bmatrix}.$$

- output of reduction: DCVP instance $(B, \mathbf{t}, d\sqrt{n+1})$
- generalization of Micciancio (2001) reduction of subset sum to CVP

DBalNPP to DCVP

DBalNPP to DCVP

- DBalNPP$_d$: Given $a_1, \ldots, a_n$ and an even number $2d > 0$, decide if a balanced partition exists with $\triangle \leq 2d$.

DBALNPP to DCVP

- DBALNPP$_d$: Given $a_1, \ldots, a_n$ and an even number $2d > 0$, decide if a balanced partition exists with $\triangle \leq 2d$. Equivalently, find $\mathbf{x} \in \{0,1\}^n$ with $\sum_j x_j = \lfloor n/2 \rfloor$ s.t. $\sum_j a_j x_j = \beta - \delta$ or $\sum_j a_j x_j = \beta + \delta$, for some $\delta \leq d$, if it exists.

DBalNPP to DCVP

- DBalNPP$_d$: Given $a_1, \ldots, a_n$ and an even number $2d > 0$, decide if a balanced partition exists with $\triangle \leq 2d$. Equivalently, find $\mathbf{x} \in \{0,1\}^n$ with $\sum_j x_j = \lfloor n/2 \rfloor$ s.t. $\sum_j a_j x_j = \beta - \delta$ or $\sum_j a_j x_j = \beta + \delta$, for some $\delta \leq d$, if it exists.

  Theorem 2. DBalNPP$_d$ *is reducible to* DCVP *for* $d > 0$.

DBalNPP to DCVP

- $\textsc{DBalNPP}_d$: Given $a_1, \ldots, a_n$ and an even number $2d > 0$, decide if a balanced partition exists with $\triangle \leq 2d$. Equivalently, find $\mathbf{x} \in \{0,1\}^n$ with $\sum_j x_j = \lfloor n/2 \rfloor$ s.t. $\sum_j a_j x_j = \beta - \delta$ or $\sum_j a_j x_j = \beta + \delta$, for some $\delta \leq d$, if it exists.

Theorem 2. $\textsc{DBalNPP}_d$ *is reducible to* DCVP *for* $d > 0$*.*

$$B' = \begin{bmatrix} 2d\, I \\ (d+1)\mathbf{1}^T \\ \mathbf{a}^T \end{bmatrix}, \quad \mathbf{t}' = \begin{bmatrix} d\,\mathbf{1} \\ (d+1)\lfloor n/2 \rfloor \\ \beta \end{bmatrix}.$$

$\mathrm{DB{\scriptsize AL}NPP}$ to DCVP

- $\mathrm{DB{\scriptsize AL}NPP}_d$: Given $a_1, \ldots, a_n$ and an even number $2d > 0$, decide if a balanced partition exists with $\triangle \leq 2d$. Equivalently, find $\mathbf{x} \in \{0,1\}^n$ with $\sum_j x_j = \lfloor n/2 \rfloor$ s.t. $\sum_j a_j x_j = \beta - \delta$ or $\sum_j a_j x_j = \beta + \delta$, for some $\delta \leq d$, if it exists.

  Theorem 2. $\mathrm{DB{\scriptsize AL}NPP}_d$ *is reducible to* DCVP *for* $d > 0$.

$$B' = \begin{bmatrix} 2d\, I \\ (d+1)\mathbf{1}^T \\ \mathbf{a}^T \end{bmatrix}, \quad \mathbf{t}' = \begin{bmatrix} d\,\mathbf{1} \\ (d+1)\lfloor n/2 \rfloor \\ \beta \end{bmatrix}.$$

- output of reduction: DCVP instance $(B', \mathbf{t}', d\sqrt{n+1})$

A lattice algorithm for NPP

A lattice algorithm for NPP

- Given a DCVP *oracle*, do a binary search on $[0, \beta]$ for $\triangle^*$

A lattice algorithm for NPP

- Given a DCVP *oracle*, do a binary search on $[0, \beta]$ for $\triangle^*$
- NPP is solved using a polynomial $\#$ calls to the oracle

A lattice algorithm for NPP

- Given a DCVP *oracle*, do a binary search on $[0, \beta]$ for $\triangle^*$
- NPP is solved using a polynomial $\#$ calls to the oracle
- **but,**

A lattice algorithm for NPP

- Given a DCVP *oracle*, do a binary search on $[0, \beta]$ for $\triangle^*$
- NPP is solved using a polynomial $\#$ calls to the oracle
- **but,** DCVP is NP-complete!

A lattice algorithm for NPP

- Given a DCVP *oracle*, do a binary search on $[0, \beta]$ for $\triangle^*$
- NPP is solved using a polynomial $\#$ calls to the oracle
- **but,** DCVP is NP-complete! no such oracle exists for large n

A lattice algorithm for NPP

- Given a DCVP *oracle*, do a binary search on $[0, \beta]$ for $\triangle^*$
- NPP is solved using a polynomial $\#$ calls to the oracle
- **but,** DCVP is NP-complete! no such oracle exists for large n
- algo does not use estimates on expected $\triangle^*$

A Basis Reduction Heuristic for NPP

A Basis Reduction Heuristic for NPP

- (try to) solve DCVP using BR on

A Basis Reduction Heuristic for NPP

- (try to) solve DCVP using BR on

$$D = \begin{bmatrix} B & \mathbf{t} \\ \mathbf{0} & M \end{bmatrix} = \begin{bmatrix} 2d\, I & d\mathbf{1} \\ \mathbf{a}^T & \beta \\ \mathbf{0} & M \end{bmatrix},$$

A Basis Reduction Heuristic for NPP

- (try to) solve DCVP using BR on

$$D = \begin{bmatrix} B & \mathbf{t} \\ \mathbf{0} & M \end{bmatrix} = \begin{bmatrix} 2d\, I & d\mathbf{1} \\ \mathbf{a}^T & \beta \\ \mathbf{0} & M \end{bmatrix}, \quad \text{where } M \text{ is a large number.}$$

A Basis Reduction Heuristic for NPP

- (try to) solve DCVP using BR on

$$D = \begin{bmatrix} B & \mathbf{t} \\ \mathbf{0} & M \end{bmatrix} = \begin{bmatrix} 2d\, I & d\mathbf{1} \\ \mathbf{a}^T & \beta \\ \mathbf{0} & M \end{bmatrix}, \quad \text{where } M \text{ is a large number.}$$

- DCVP $\rightarrow$ shortest vector problem (SVP); Kannan (87)

A Basis Reduction Heuristic for NPP

- (try to) solve DCVP using BR on

$$D = \begin{bmatrix} B & \mathbf{t} \\ \mathbf{0} & M \end{bmatrix} = \begin{bmatrix} 2d\,I & d\mathbf{1} \\ \mathbf{a}^T & \beta \\ \mathbf{0} & M \end{bmatrix}, \quad \text{where } M \text{ is a large number.}$$

- $\mathrm{DCVP} \rightarrow$ shortest vector problem (SVP); Kannan (87)

- with $\triangle^* = \sqrt{n}\,2^{-n}\,R$, try $d = c\triangle^*$ for several c's in $[1/n, n]$

A Basis Reduction Heuristic for NPP

- (try to) solve DCVP using BR on

$$D = \begin{bmatrix} B & \mathbf{t} \\ \mathbf{0} & M \end{bmatrix} = \begin{bmatrix} 2d\,I & d\mathbf{1} \\ \mathbf{a}^T & \beta \\ \mathbf{0} & M \end{bmatrix}, \quad \text{where } M \text{ is a large number.}$$

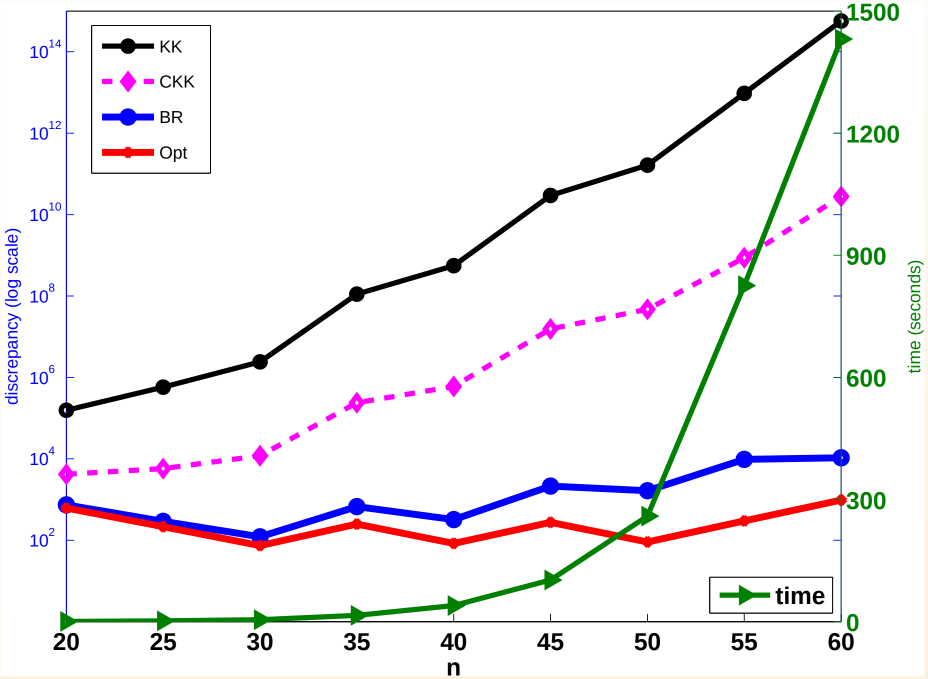
- $\mathrm{DCVP} \rightarrow$ shortest vector problem (SVP); Kannan (87)

- with $\triangle^* = \sqrt{n}\,2^{-n}\,R$, try $d = c\triangle^*$ for several c's in $[1/n, n]$

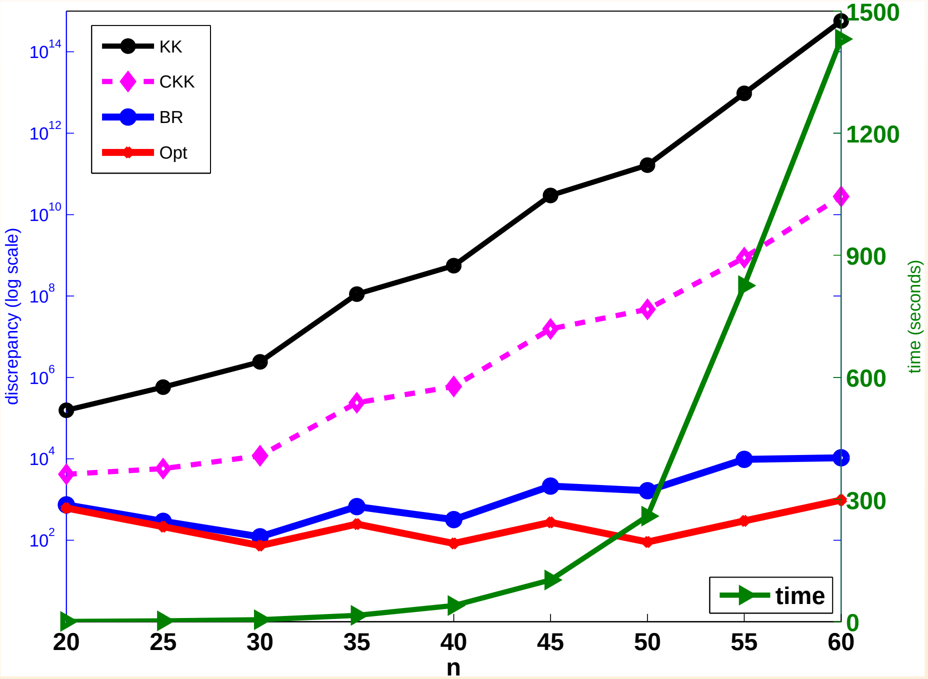
- Lagarias & Odlyzko (85), Coster et al. (92): for subset sums

BR Algo Tests:

BR Algo Tests: for NPP

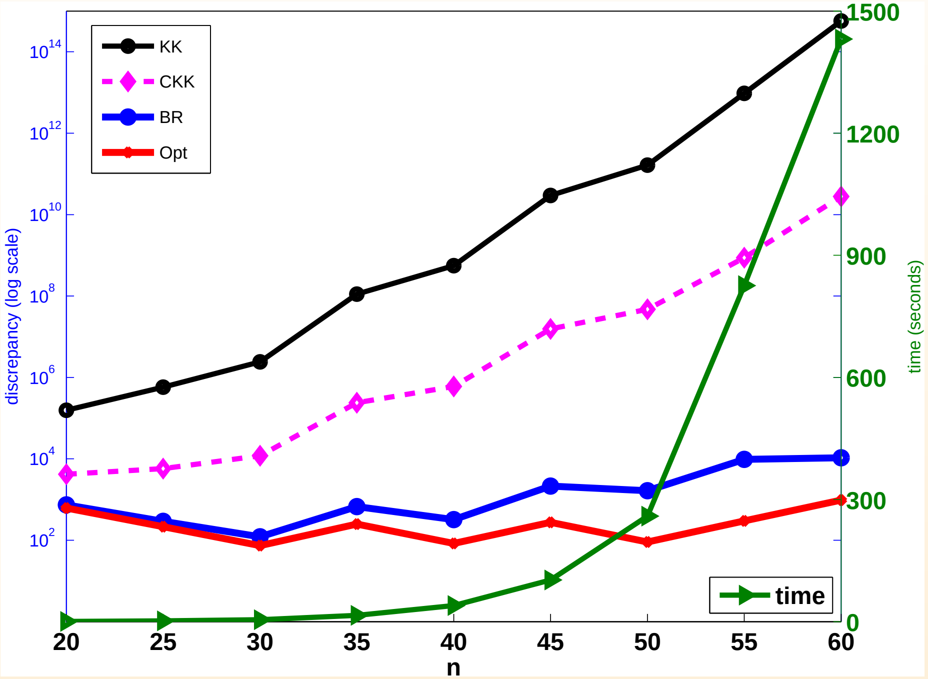


BR Algo Tests: for NPP



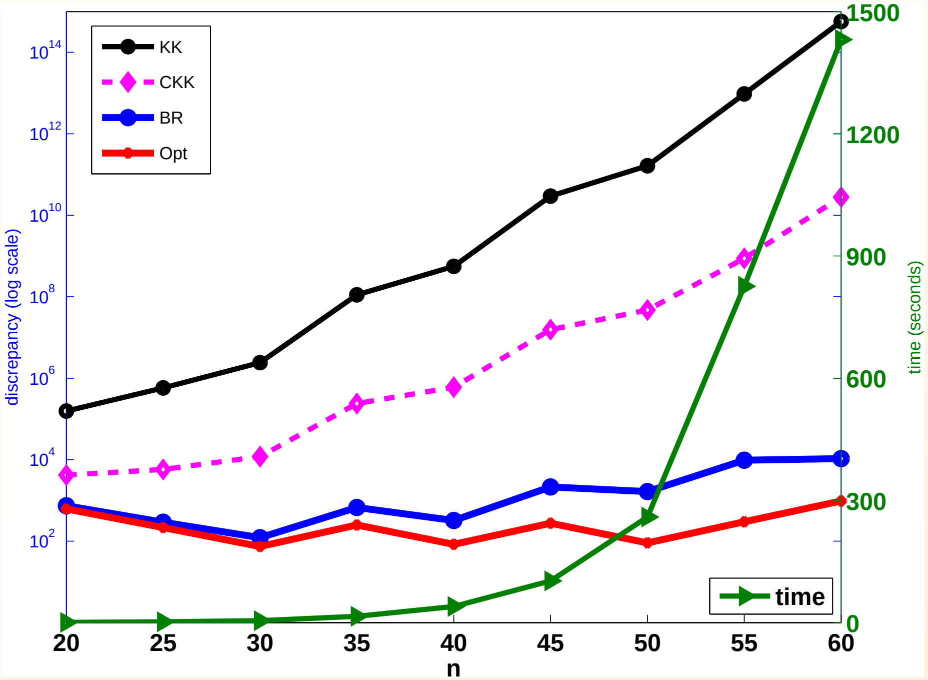
- block Korkine-Zolotarev (BKZ) reduction

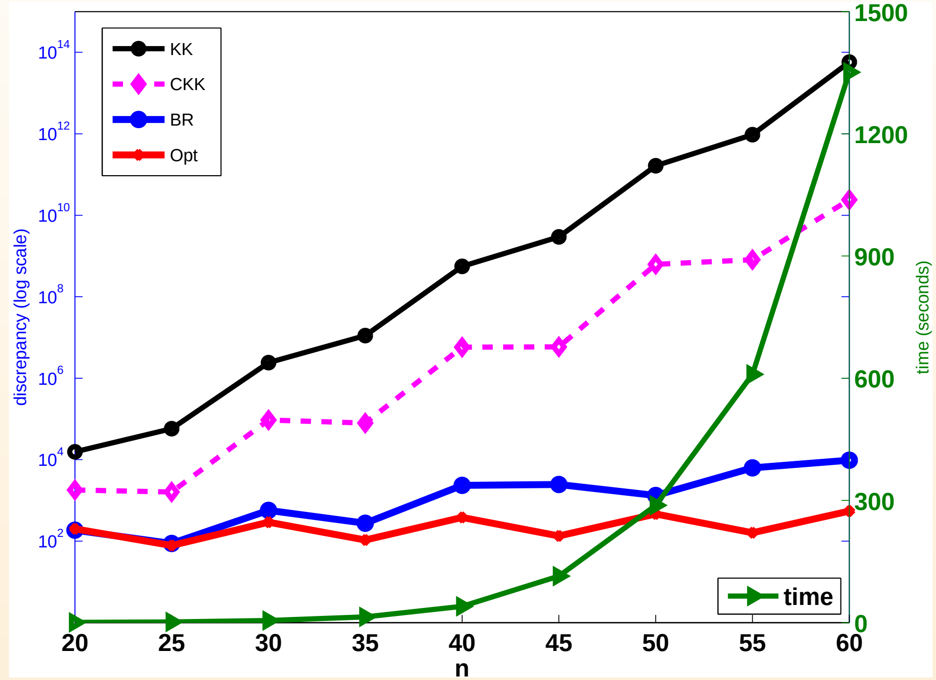
BR Algo Tests: for NPP



- block Korkine-Zolotarev (BKZ) reduction
- opt: $\triangle^* = \sqrt{n}\, 2^{-n} R$ is plotted
- ckk: estimated $\triangle_{CKK}$ for same running time as BR

BR Algo Tests: for NPP and BALNPP





- block Korkine-Zolotarev (BKZ) reduction
- opt: $\triangle^* = \sqrt{n}\, 2^{-n}\, R$ is plotted
- ckk: estimated $\triangle_{CKK}$ for same running time as BR

Mixed Integer Program (MIP) for NPP

Mixed Integer Program (MIP) for NPP

- let $x_j = 1$ if a_j is put in first subset, and 0 otherwise; and $w =$ deviation from perfect division for first subset.

Mixed Integer Program (MIP) for NPP

- let $x_j = 1$ if a_j is put in first subset, and 0 otherwise; and
  $w =$ deviation from perfect division for first subset.
  Discrepancy $\triangle = 2w$.

Mixed Integer Program (MIP) for NPP

- let $x_j = 1$ if a_j is put in first subset, and 0 otherwise; and
  $w =$ deviation from perfect division for first subset.

  Discrepancy $\triangle = 2w$.

  MIP for NPP:

$$\begin{array}{llcll}
\min & 2w & & & \\
\text{s.t.} & w & \geq & \sum a_j x_j - \beta & \\
 & w & \geq & -\sum a_j x_j + \beta & \\
 & x_j & \in & \{0,1\} & j = 1, \ldots, n.
\end{array}$$

CBR-R for NPP MIP

CBR-R for NPP MIP

- write NPP MIP as $\min\{\, w \mid A\mathbf{x} + Bw \le \mathbf{b},\ \mathbf{x} \in \mathbb{Z}^n \,\}$ with

$$A = \begin{bmatrix} \mathbf{a}^T \\ -\mathbf{a}^T \\ -I \\ I \end{bmatrix}, \quad B = \begin{bmatrix} -1 \\ -1 \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} \beta \\ -\beta \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix};$$

CBR-R for NPP MIP

- write NPP MIP as $\min\{\, w \mid A\mathbf{x} + Bw \leq \mathbf{b},\ \mathbf{x} \in \mathbb{Z}^n \,\}$ with

$$A = \begin{bmatrix} \mathbf{a}^T \\ -\mathbf{a}^T \\ -I \\ I \end{bmatrix}, \quad B = \begin{bmatrix} -1 \\ -1 \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} \beta \\ -\beta \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix};$$

  apply basis reduction on $D = \begin{bmatrix} A & \mathbf{b} \\ \mathbf{0} & M \end{bmatrix}$ to obtain $\tilde{D} = \begin{bmatrix} \tilde{A} & \tilde{\mathbf{b}} \\ \mathbf{0} & M \end{bmatrix}$,

CBR-R for NPP MIP

- write NPP MIP as $\min\{\, w \mid A\mathbf{x} + Bw \le \mathbf{b},\ \mathbf{x} \in \mathbb{Z}^n \,\}$ with

$$A = \begin{bmatrix} \mathbf{a}^T \\ -\mathbf{a}^T \\ -I \\ I \end{bmatrix}, \quad B = \begin{bmatrix} -1 \\ -1 \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} \beta \\ -\beta \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix};$$

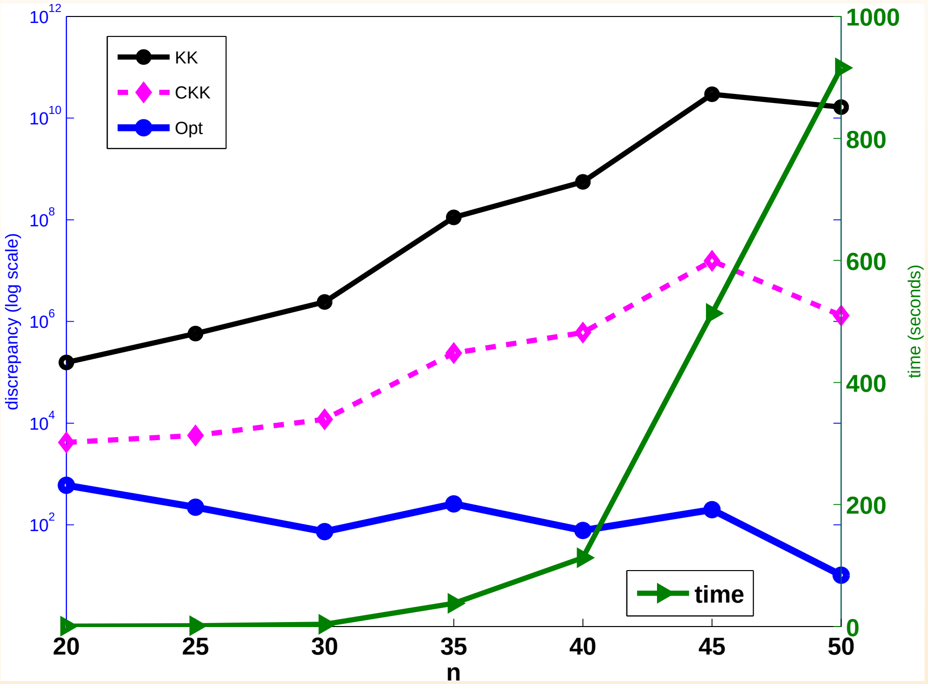
apply basis reduction on $D = \begin{bmatrix} A & \mathbf{b} \\ \mathbf{0} & M \end{bmatrix}$ to obtain $\tilde{D} = \begin{bmatrix} \tilde{A} & \tilde{\mathbf{b}} \\ \mathbf{0} & M \end{bmatrix}$,

- solve the CBR-R reformulation using standard solver:

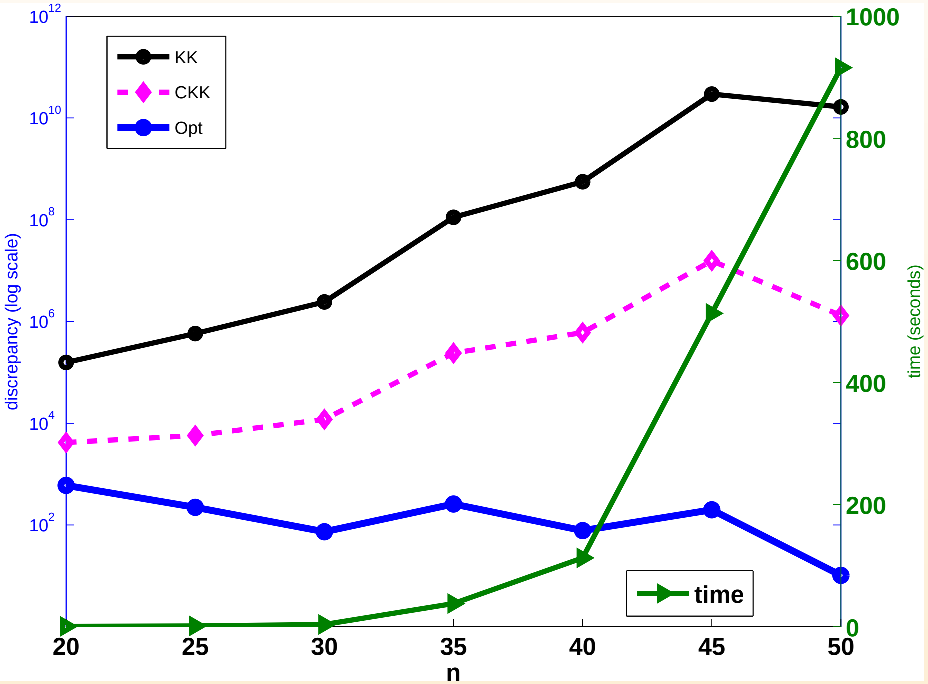
$$\min\{\, w \mid \tilde{A}\mathbf{y} + Bw \le \tilde{\mathbf{b}},\ \mathbf{y} \in \mathbb{Z}^n \,\}$$

CBR-R Tests:

CBR-R Tests: on NPP

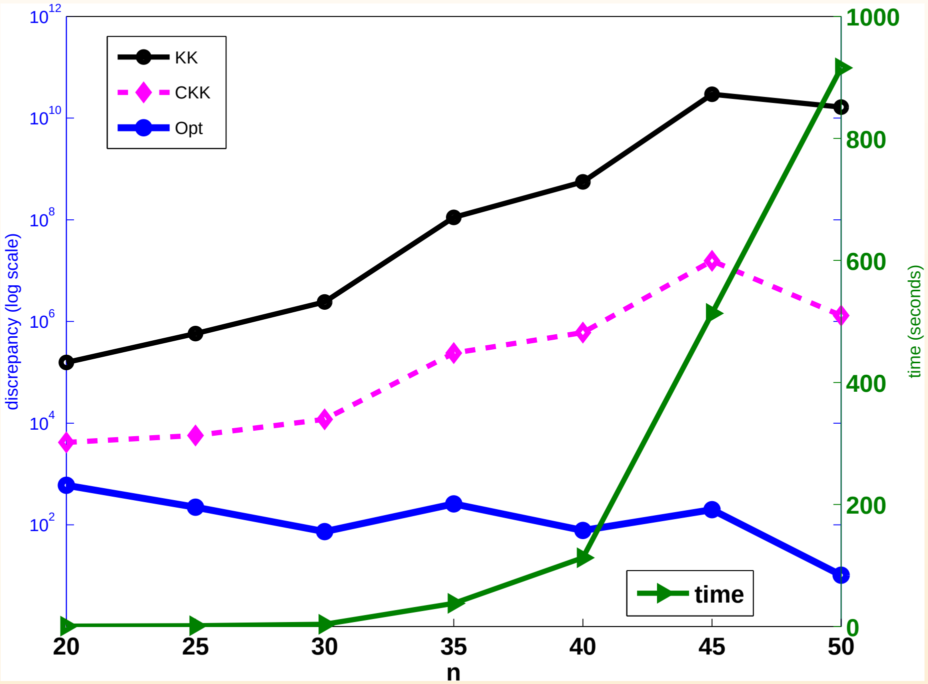


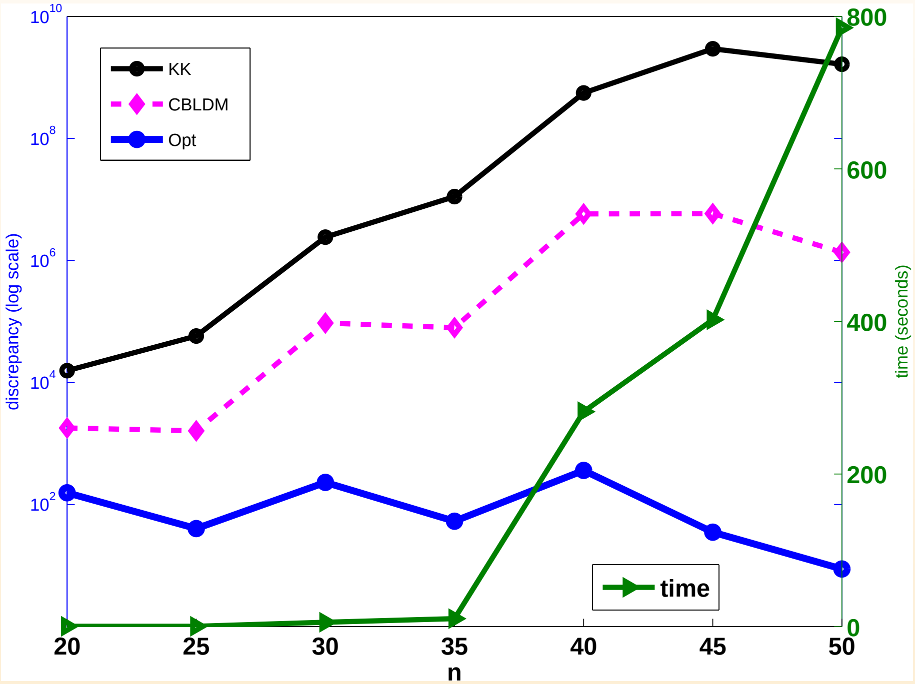
CBR-R Tests: on NPP



- BKZ for BR, CPLEX 9.0 as MIP solver

CBR-R Tests: on NPP and BalNPP





- BKZ for BR, CPLEX 9.0 as MIP solver

Remarks

Remarks

- BR, CBR-R: can be applied to

Remarks

- BR, CBR-R: can be applied to
  - unequal partitions (e.g., $\beta = 0.3\alpha$)
  - *constrained* partitions ($\sum_j x_j = r \neq n/2$)
  - NPP with $k \geq 3$ subsets, with unequal shares ($\neq 1/k$), and/or cardinality constraints

Remarks

- BR, CBR-R: can be applied to
  - unequal partitions (e.g., $\beta = 0.3\alpha$)
  - *constrained* partitions ($\sum_j x_j = r \neq n/2$)
  - NPP with $k \geq 3$ subsets, with unequal shares ($\neq 1/k$), and/or cardinality constraints
- lattice algos efficient in practice for reasonably large n

Remarks

- BR, CBR-R: can be applied to
  - unequal partitions (e.g., $\beta = 0.3\alpha$)
  - *constrained* partitions ($\sum_j x_j = r \neq n/2$)
  - NPP with $k \geq 3$ subsets, with unequal shares ($\neq 1/k$), and/or cardinality constraints

- lattice algos efficient in practice for reasonably large n

- running times increase with R

Remarks

- BR, CBR-R: can be applied to
  - unequal partitions (e.g., $\beta = 0.3\alpha$)
  - *constrained* partitions ($\sum_j x_j = r \neq n/2$)
  - NPP with $k \geq 3$ subsets, with unequal shares ($\neq 1/k$), and/or cardinality constraints
- lattice algos efficient in practice for reasonably large n
- running times increase with R
- for $n \geq 100$, KK may still be the best (current) option

Outline

- Number Partitioning Problem (NPP)
- Karmarkar-Karp differencing (KK)
- NPP and the Closest Vector Problem (CVP)
- A Basis Reduction Heuristic for NPP
- Mixed Integer Program (MIP) for NPP