Title	A Combinatorial Problem Arising from Polyhedral Homotopies for Solving Polynomial Systems (Mathematical Science of Optimization)
Author(s)	Takeda, Akiko; Kojima, Masakazu; Fujisawa, Katsuki
Citation	数理解析研究所講究録 (2000), 1174: 146-158
Issue Date	2000-10
URL	http://hdl.handle.net/2433/64464
Right	
Type	Departmental Bulletin Paper
Textversion	publisher

数理解析研究所講究録
1174巻 2000年 146-158

A Combinatorial Problem Arising from Polyhedral Homotopies for Solving Polynomial Systems

[1] 武田 朗子, (Akiko Takeda)
[1] 小島 政和, (Masakazu Kojima)
[2] 藤澤 克樹 (Katsuki Fujisawa)

[1] 東京工業大学 数理・計算科学専攻
(Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology)

[2] 京都大学 建築学専攻 (Dept. of Architecture, Kyoto University)

1 Introduction.

Homotopy continuation methods have been developed as efficient numerical algorithms for solving all isolated zeros of polynomial systems. Among others, a new type of homotopy method has emerged in the last few years. Let $P(\boldsymbol{x}) = (p_1(\boldsymbol{x}), \ldots, p_n(\boldsymbol{x})) = \boldsymbol{0}$ be a system of n polynomial equations in n unknowns. Each polynomial consists of more than one monomials. We denote each monomial as $\boldsymbol{x}^{\boldsymbol{a}} = x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n}$ and identify it with a lattice point $\boldsymbol{a} = (a_1, a_2, \ldots, a_n) \in N^n \equiv \{0, 1, 2, \ldots\}^n$. Then, denoting each term of the ith equation as $(c_{i,a}\ \boldsymbol{x}^{\boldsymbol{a}})$ with a complex coefficient $c_{i,a} \in C$ and $\boldsymbol{a} \in N^n$, we write a system of polynomials $P(\boldsymbol{x}) = (p_1(\boldsymbol{x}), \ldots, p_n(\boldsymbol{x}))$ as

$$p_i(\boldsymbol{x}) = \sum_{\boldsymbol{a} \in \mathcal{A}_i} c_{i,a}\ \boldsymbol{x}^{\boldsymbol{a}}, \quad i = 1, \ldots, n. \tag{1}$$

Here, $\mathcal{A}_i$ $(i = 1, \ldots, n)$ is a set of lattice points corresponding to terms of the ith equation, and is called the *support* of the polynomial $p_i(\boldsymbol{x})$. Homotopy continuation methods for solving $P(\boldsymbol{x}) = \boldsymbol{0}$ construct trivial systems $Q(\boldsymbol{x}) = (q_1(\boldsymbol{x}), \ldots, q_n(\boldsymbol{x}))$ and homotopy systems $H(\boldsymbol{x}, t)$ at the beginning. If these systems $Q(\boldsymbol{x})$ and $H(\boldsymbol{x}, t)$ are chosen correctly, the following three properties hold:

Property 0: The solutions of $Q(\boldsymbol{x}) = \boldsymbol{0}$ are known.

Property 1: The solution set of $H(\boldsymbol{x}, t) = \boldsymbol{0}$ for $0 \leq t \leq 1$ consists of a finite number of smooth (homotopy) paths, each parameterized by t in $[0, 1)$.

Property 2: $H(\boldsymbol{x}, t)$ satisfies $H(\boldsymbol{x}, 1) = P(\boldsymbol{x})$ and $H(\boldsymbol{x}, 0) = Q(\boldsymbol{x})$.

When the three properties hold, solution paths of $H(\boldsymbol{x}, t) = \boldsymbol{0}$ can be followed from initial solutions of $Q(\boldsymbol{x}) = \boldsymbol{0}$ at $t = 0$ to all solutions of the original problem $P(\boldsymbol{x}) = \boldsymbol{0}$ at $t = 1$, using standard numerical techniques. It has been shown theoretically and also experimentally [5, 6] that the so called *polyhedral homotopy* [5] generates much fewer homotopy paths than classical linear homotopy methods. Since the polyhedral homotopy method was proposed, the study related to efficient computation of mixed cells has been progressing. The process of locating all mixed cells is necessary to define start systems $Q(\boldsymbol{x}) = \boldsymbol{0}$ and homotopies $H(\boldsymbol{x}, t) = \boldsymbol{0}$ which satisfy Properties 0,1 and 2, but it occupies the majority of computation in the polyhedral homotopy method. Our contribution lies in developing an algorithm for mixed cell computation with a clever use of linear programming techniques and implementing the algorithm on a parallel computing system. As our numerical results show, we are now able to deal with larger dimensional polynomial systems with less computational time.

This paper is organized as follows. Section 2 introduces our problem along with new terminology related to mixed cell computation. Section 3 presents a new approach for enumerating all solutions

of our problem. The approach induces a family of linear-inequality systems, which we reformulate via linear programs in Section 4. Section 5 shows our sequential algorithm and its parallel version, and then, examines both algorithms on standard benchmark problems comparing them to other mixed cell implementations. This paper concludes with a summary and directions for future work in Section 6.

2 An enumeration problem in polyhedral homotopies.

According to the paper [5], we distinguish three cases of the polynomial system (1): The input system (1) is *unmixed* when all the sets $\mathcal{A}_i$ $(i = 1, \ldots, n)$ are equal; *fully mixed* when they are all distinct; and *semimixed* when they are equal in m distinct blocks. In this section, we will concentrate on semimixed systems; mixed and unmixed systems are special cases.

For $j = 1, \ldots, m$, let s_j be the number of polynomials in $P(\boldsymbol{x})$ having the support $\mathcal{A}_i$. Note that m is a positive integer not greater than n, and s_j $(j = 1, \ldots, m)$ are positive integers such that $\sum_{j=1}^{m} s_j = n$ and $s_j \geq 1$. To simplify the notation, let M denote the set of indices $\{1, 2, \ldots, m\}$, $\boldsymbol{\beta} = (\beta_j : j \in M)$ a m dimensional vector, and $\langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle$ the inner product of two vectors $\boldsymbol{a}$ and $\boldsymbol{\alpha}$ in the n-dimensional Euclidean space R^n. Also, let $\#C$ denote the number of elements.

Problem 2.1. (Semimixed polynomial system)
Let $\omega_j(\boldsymbol{a})$ be a real number chosen generically for $\forall \boldsymbol{a} \in \mathcal{A}_j$ and $\forall j \in M$. Find a solution $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in R^{n+m}$ which satisfies

$$\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a}), \quad \forall \boldsymbol{a} \in \mathcal{A}_j, \ \forall j \in M, \tag{2}$$

with exactly $(s_j + 1)$ equalities for each $j \in M$.

Since each $\omega_j(\boldsymbol{a})$ is chosen generically for $\forall \boldsymbol{a} \in \mathcal{A}_j$ and $\forall j \in M$, we may assume the following condition on Problem 2.1.

Condition 2.2. (Nondegeneracy condition) Let $\emptyset \neq K \subseteq M$, $F_j \subseteq \mathcal{A}_j$ $(j \in K)$ and $\ell = \sum_{j \in K} \#F_j > n + \#K$. Then the system of ℓ linear equations with $(n + \#K)$ variables $\boldsymbol{\alpha}$ and β_j $(j \in K)$ such that $\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a})$ $(\boldsymbol{a} \in F_j, \ j \in K)$, has no solution.

Suppose that for some cell $(C_1, \ldots, C_m)$ such that $C_j \subseteq \mathcal{A}_j$ and $\#C_j = s_j + 1$ $(j = 1, \ldots, m)$, a solution $(\widetilde{\boldsymbol{\alpha}}, \widetilde{\boldsymbol{\beta}}) \in R^{n+m}$ of Problem 2.1 satisfies

$$\begin{aligned} \widetilde{\beta_j} - \langle \boldsymbol{a}, \widetilde{\boldsymbol{\alpha}} \rangle &= \omega_j(\boldsymbol{a}) \quad \text{for } \forall \boldsymbol{a} \in C_j, \ \forall j \in M, \\ \widetilde{\beta_j} - \langle \boldsymbol{a}, \widetilde{\boldsymbol{\alpha}} \rangle &< \omega_j(\boldsymbol{a}) \quad \text{for } \forall \boldsymbol{a} \in \mathcal{A}_j \setminus C_j, \ \forall j \in M. \end{aligned}$$

Then, the cell $C = (C_1, \ldots, C_m)$ is called a *fine mixed cell* of $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m)$, induced by the lifting $\omega = (\omega_1, \ldots, \omega_n)$. Here, we denote an $(s_j + 1)$-element subset $C_j \subseteq \mathcal{A}_j$ by $C_j = \{\boldsymbol{a}_0^j, \boldsymbol{a}_1^j, \ldots, \boldsymbol{a}_{s_j}^j\}$. Condition 2.2 further ensures that the set of $\sum_{j \in M} (s_j + 1)$ points

$$\left\{ \begin{pmatrix} \boldsymbol{a}_p^j \\ \boldsymbol{e}^j \end{pmatrix} \in N^{n+m} : p = 0, 1, 2, \ldots, s_j, \ j \in M \right\}$$

is linearly independent, where $\boldsymbol{e}^j$ denotes the jth unit vector in R^m. It follows in particular that the set of $(s_j + 1)$ points $\{\boldsymbol{a}_p^j : p = 0, 1, 2, \ldots, s_j\}$ induces an s_j-dimensional simplex in R^n, and that the $n \times n$ matrix $\boldsymbol{A}$ defined by

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_1 \\ \boldsymbol{A}_2 \\ \cdots \\ \boldsymbol{A}_m \end{pmatrix}, \ \boldsymbol{A}_j = \begin{pmatrix} \boldsymbol{a}_1^j - \boldsymbol{a}_0^j \\ \boldsymbol{a}_2^j - \boldsymbol{a}_0^j \\ \cdots \\ \boldsymbol{a}_{s_j}^j - \boldsymbol{a}_0^j, \end{pmatrix} \in R^{s_j \times n}. \tag{3}$$

is nonsingular.

A collection of all fine mixed cells, which follow from all solutions of Problem 2.1, is called a *fine mixed subdivision* of $\mathcal{A}$. Assuming that the number of all solutions for Problem 2.1 is r, $C^{(j)} = (C_1^{(j)}, \ldots, C_m^{(j)})$ $(j = 1, \ldots, r)$ are all mixed cells, and $\mathcal{S} = \{C^{(1)}, \ldots, C^{(r)}\}$ is a fine mixed subdivision of $\mathcal{A}$. Then, r nonsingular matrices $\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(r)}$ are induced from fine mixed cells $C^{(j)}$ $(j = 1, \ldots, r)$. In the polyhedral homotopy continuation method, it is a key issue how efficiently we can obtain a fine mixed subdivision $\mathcal{S}$ of $\mathcal{A}$, that is, **all solutions** of Problem 2.1. For more details about relationship between a fine mixed subdivision and the polyhedral homotopy method, see the articles [3, 4, 5, 6, 7, 11, etc].

We present a special case of Problem 2.1 with $m = n$ and $(s_1, \ldots, s_m) = (1, \ldots, 1)$. In this case, the set M is defined as $M = \{1, 2, \ldots, n\}$.

Problem 2.1' (Fully mixed polynomial system)
Let $\omega_j(\boldsymbol{a})$ be a real number chosen generically for $\forall \boldsymbol{a} \in \mathcal{A}_j$ and $\forall j \in M$. Find a solution $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in R^{2n}$ which satisfies $\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a})$, $\forall \boldsymbol{a} \in \mathcal{A}_j$, $\forall j \in M$, with exactly 2 equalities for each $j \in M$.

3 Systems embedded in an enumeration tree.

One easy solution method for Problem 2.1 is as follows; At first, prepare the set

$$\widetilde{\mathcal{S}} = \left\{ \boldsymbol{C} = (C_1, \ldots, C_m) : \quad C_j \subseteq \mathcal{A}_j, \quad \sharp C_j = s_j + 1 \ (j \in M) \right\},$$

and solve the equality system

$$\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a}), \quad \forall \boldsymbol{a} \in C_j, \ \forall j \in M, \tag{4}$$

for $\forall \boldsymbol{C} = (C_1, \ldots, C_m) \in \widetilde{\mathcal{S}}$. And then, check the feasibility of the solution $(\bar{\boldsymbol{\alpha}}, \bar{\boldsymbol{\beta}}) \in R^{n+m}$ of (4) for the remaining linear inequalities such as $\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a})$, $\forall \boldsymbol{a} \in \mathcal{A}_j \setminus C_j$, $\forall j \in M$. This method becomes impractical when the given polynomial system (1) has relatively many monomials, *i.e.*, $\mathcal{A}_j$ includes a large number of elements $(j \in M)$. Then, $\widetilde{\mathcal{S}}$ consists of many elements $\boldsymbol{C} = (C_1, \ldots, C_m)$, and tremendously many linear systems of (4) are prepared. In this section, we introduce a practical enumeration technique to avoid such exhaustive feasibility tests for all possible linear systems.

3.1 A family of systems of linear inequalities.

For each nonempty $K \subseteq M$ and each $\boldsymbol{F} = (F_j : j \in K)$ with $F_j \subseteq \mathcal{A}_j$, we consider a system which consists of equalities and inequalities such as

$$\mathrm{E}(K, \boldsymbol{F}): \left\{ \begin{array}{l} \beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a}) \ \ (\boldsymbol{a} \in F_j, \ j \in K), \\ \beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a}) \ \ (\boldsymbol{a} \in \mathcal{A}_j \backslash F_j, \ j \in K). \end{array} \right.$$

For every nonempty $K \subseteq M$, let

$$\left. \begin{array}{rcl} \mathcal{F}^{\leq}(K) & = & \{\boldsymbol{F} = (F_j : j \in K) \ : \ F_j \subseteq \mathcal{A}_j, \ \sharp F_j \leq s_j + 1 \ (j \in K)\}, \\ \mathcal{F}^{=}(K) & = & \{\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{\leq}(K) \ : \ \sharp F_j = s_j + 1 \ (j \in K)\}, \\ \mathcal{F}^{=\&\mathrm{f}}(K) & = & \{\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{=}(K) \ : \ \mathrm{E}(K, \boldsymbol{F}) \text{ is feasible}\}, \end{array} \right\} \tag{5}$$

and assume that $\mathcal{F}^{\leq}(\emptyset) = \mathcal{F}^{=}(\emptyset) = \mathcal{F}^{=\&\mathrm{f}}(\emptyset) = \{\emptyset\}$ for convention. From the definition of (5), we see that $\mathcal{F}^{=\&\mathrm{f}}(K) \subseteq \mathcal{F}^{=}(K) \subseteq \mathcal{F}^{\leq}(K)$ for every $K \subseteq M$. Then, computing all solutions of $\mathrm{E}(M, \boldsymbol{F})$ for all $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$ reduces to finding all solutions of Problem 2.1, *i.e.*, finding all vertices $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of the polyhedral set V with exactly $(s_j + 1)$ equalities in (2) for each $j \in M$. Condition 2.2 ensures that the

solution set of each $\mathrm{E}(M, \boldsymbol{F})$ ($\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$) consists of a single vertex of V. We will embed a tree structure in a subfamily of $\mathcal{F}^{=}(K)$ for $\forall K \subseteq M$.

Note that partial constraints of Problems 2.1 such that

$$\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a}) \ \ (\boldsymbol{a} \in \mathcal{A}_j, \ j \in M \backslash K), \tag{6}$$

are not included in $E(K, \boldsymbol{F})$. Those constraints make the feasible region of $(\boldsymbol{\alpha}, \beta_j : j \in K)$ defined by $E(K, \boldsymbol{F})$ narrow, but never change the feasibility of $E(K, \boldsymbol{F})$ for $\forall \boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K)$. That is, the set of linear inequalities (6) is irrelevant to the feasibility of $\mathrm{E}(K, \boldsymbol{F})$. This fact is obvious, since if $\mathrm{E}(K, \boldsymbol{F})$ has a feasible solution $(\widetilde{\boldsymbol{\alpha}}, \widetilde{\beta}_j : j \in K)$, then we can define $\widetilde{\beta}_j = \min\{\langle \boldsymbol{a}, \widetilde{\boldsymbol{\alpha}} \rangle + \omega_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j\}$ for $\forall j \in M \setminus K$ and the extended solution $(\widetilde{\boldsymbol{\alpha}}, \widetilde{\beta}_j : j \in M)$ satisfies constraints of (6).

Note that the set $\mathcal{F}^{=\&\mathrm{f}}(M)$ is coincident with a mixed subdivision $\mathcal{S}$ of $\mathcal{A}$, defined in the previous section. Thus, we want to enumerate all $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$ avoiding the brutal exhausting search such as feasibility tests of $\mathrm{E}(M, \boldsymbol{F})$ for $\forall \boldsymbol{F} \in \mathcal{F}^{=}(M)$. The lemma below plays an essential role to reduce the number of feasibility tests.

Lemma 3.1. *Let $\bar{K} \subseteq K \subseteq M$, $\bar{\boldsymbol{F}} \in \mathcal{F}^{\leq}(\bar{K})$, $\boldsymbol{F} \in \mathcal{F}^{\leq}(K)$, and $\bar{F}_j \subseteq F_j$ $(j \in \bar{K})$. If $E(\bar{K}, \bar{\boldsymbol{F}})$ is infeasible, then so is $E(K, \boldsymbol{F})$.*

Proof: All constraints of $\mathrm{E}(\bar{K}, \bar{\boldsymbol{F}})$ are included in $\mathrm{E}(K, \boldsymbol{F})$. Therefore, if $\mathrm{E}(\bar{K}, \bar{\boldsymbol{F}})$ has no feasible solution, then $\mathrm{E}(K, \boldsymbol{F})$ does. ∎

This lemma implies that if we find that $\mathrm{E}(\bar{K}, \bar{\boldsymbol{F}})$ with $\bar{\boldsymbol{F}} \in \mathcal{F}^{=}(\bar{K})$ is infeasible, we can omit feasibility checks for $\mathrm{E}(M, \boldsymbol{F})$ with $\boldsymbol{F} \in \mathcal{F}^{=}(M)$ satisfying $F_j = \bar{F}_j$ $(j \in \bar{K})$. Hence, the test described in Lemma 3.1 will save computation for solving many linear systems $\mathrm{E}(M, \boldsymbol{F})$ with $\boldsymbol{F} \in \mathcal{F}^{=}(M)$, especially when $\#\bar{K}$ is much less than $\#M(= m)$.

3.2 Embedding a tree structure.

Let $K_0, K_1, K_2, \ldots, K_\ell$ be distinct subsets of M such that $\emptyset = K_0 \subset K_1 \subset K_2 \subset \ldots \subset K_\ell = M$. Typically we take

$$\ell = m, \ K_0 = \emptyset, \ K_p = \{1, 2, \ldots, p\} \ \ (p = 1, 2, \ldots, m). \tag{7}$$

In order to enumerate all $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$, we build a tree structure into the family of subsets $\boldsymbol{F} \in \mathcal{F}^{=}(K_p)$ $(p = 0, 1, 2, \ldots, \ell)$. We first place the empty set $\mathcal{F}^{=}(K_0)$ at the root node. For each $p = 1, 2, \ldots, \ell$, we then place the sets $\boldsymbol{F} \in \mathcal{F}^{=}(K_p)$ in the pth level of the tree that we are building. A node $\boldsymbol{F}' \in \mathcal{F}^{=}(K_{p'})$ in the p'th level with $p' > p$ is a *descendant* of a node $\boldsymbol{F} \in \mathcal{F}^{=}(K_p)$ in the pth level if and only if $F'_j = F_j$ for $\forall j \in K_p$. In the special case that $p' = p + 1$, the descendant $\boldsymbol{F}' \in \mathcal{F}^{=}(K_{p'})$ is called *child* for a node $\boldsymbol{F} \in \mathcal{F}^{=}(K_p)$. Specifically, all $\boldsymbol{F} \in \mathcal{F}^{=}(K_1)$ are child nodes of the root node $\mathcal{F}^{=}(K_0)$, and the nodes $\boldsymbol{F} \in \mathcal{F}^{=}(M)$ in the ℓth level are leaf nodes having no child nodes.

Now we are ready to describe a basic framework of our method for enumerating all the nodes in $\mathcal{F}^{=\&\mathrm{f}}(M)$. From the root node $\emptyset \in \mathcal{F}^{=}(K_0)$, we will apply the depth-first search to the tree structure that we have built above. In usual cases, $E(\emptyset, K_0)$ associated with the root node is feasible, and we go down to one of its child nodes $\boldsymbol{F} \in \mathcal{F}^{=}(K_1)$. At each node $\boldsymbol{F} \in \mathcal{F}^{=}(K_p)$, we check whether $\mathrm{E}(K_p, \boldsymbol{F})$ is feasible, *i.e.*, $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}$, by using the simplex method for a linear program related to $\mathrm{E}(K_p, \boldsymbol{F})$. More technical details of this part will be described later. If $\mathrm{E}(K_p, \boldsymbol{F})$ is infeasible, then all of its descendants are found to be infeasible from Lemma 3.1; hence its descendants never contain any node in $\mathcal{F}^{=\&\mathrm{f}}(M)$. In this case, we can terminate the node $\boldsymbol{F} \in \mathcal{F}^{=}(K_p)$ in the pth level, and we will backtrack the tree.

On the other hand, if the problem $\mathrm{E}(K_p, \boldsymbol{F})$ is feasible, *i.e.*, $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K_p)$, and $p < \ell$, we will go down the tree to one of its child node. If $\mathrm{E}(K_p, \boldsymbol{F})$ is feasible and $p = \ell$, then we obtain one

desired node in $\mathcal{F}^{=\&\mathrm{f}}(M)$. Proceeding this enumeration procedure, we eventually generate all nodes in $\cup_{p=1}^{\ell}\mathcal{F}^{=\&\mathrm{f}}(K_p)$. It should be noted that these nodes induce a subtree of the entire tree that we have built above.

It should be emphasized that we can easily adapt this framework to parallel computation. Define $K_0, K_1, \ldots, K_\ell$ as in (7). Taking some $p \geq 1$, consider the pth family of linear-inequality systems $\mathrm{E}(K_p, \boldsymbol{F})$ $(\forall \boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K_p))$. Then the number of systems in the pth family amounts to $\#\mathcal{F}^{=\&\mathrm{f}}(K_p)$. We allocate all linear-inequality systems $\mathrm{E}(K_p, \boldsymbol{F})$ $(\forall \boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K_p))$ among multiple processors. On each processor, apply the depth first search described above to a subtree whose root node corresponds to the assigned node $\widetilde{\boldsymbol{F}} \in \mathcal{F}^{=\&\mathrm{f}}(K_p)$, and finally, compute solutions $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in R^{n+m}$ of $\mathrm{E}(M, \boldsymbol{F})$ for all descendants $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$ of the root node $\widetilde{\boldsymbol{F}}$. Summing up all solutions from each processor, we obtain all solutions of Problem 2.1. We need to take the number of systems in the pth family (that is, $\#\mathcal{F}^{=\&\mathrm{f}}(K_p)$) into account by choosing the number p carefully, in order to balance the number of systems $\#\mathcal{F}^{=\&\mathrm{f}}(K_p)$ with the number of available processors.

4 Reformulation via linear programs

Corresponding to each system $\mathrm{E}(K, \boldsymbol{F})$ with $K \subseteq M$ and $\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{\leq}(K)$, we consider a linear program:

$$
\begin{array}{lll}
\mathrm{P}(K, \boldsymbol{F}): & \text{maximize} & \displaystyle\sum_{j \in K} b_j \beta_j - \langle \boldsymbol{d}, \boldsymbol{\alpha} \rangle \\
& \text{subject to} & \beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in F_j,\ j \in K), \\
& & \beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in \mathcal{A}_j \backslash F_j,\ j \in K).
\end{array}
$$

Note that the constraint linear inequalities of $\mathrm{P}(K, \boldsymbol{F})$ coincide with those appeared in $\mathrm{E}(K, \boldsymbol{F})$. Hence, the system of linear inequalities $\mathrm{E}(K, \boldsymbol{F})$ is feasible if and only if the problem $\mathrm{P}(K, \boldsymbol{F})$ is feasible, and each solution of Problem 2.1 is corresponding to a solution of $\mathrm{P}(M, \boldsymbol{F})$ with some $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$ and vice versa. In order to apply the duality theory between $\mathrm{P}(K, \boldsymbol{F})$ and its dual program effectively, we need to make the dual program feasible by choosing $\boldsymbol{d}$ and b_j $(j \in K)$ of the objective function appropriately, though we can take any linear function as the objective function in $\mathrm{P}(K, \boldsymbol{F})$. Also, in Section 4.3, we will show how to update these $\boldsymbol{d}$ and b_j $(j \in K_p)$ when we proceed from $\mathrm{P}(K_p, \boldsymbol{F})$ with $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K_p)$ in the pth level to a child node $\mathrm{P}(K_{p+1}, \boldsymbol{F}')$ with $\boldsymbol{F}' \in \mathcal{F}^{\leq}(K_{p+1})$ in the $(p+1)$st level.

In the following subsections, we will show feasibility and infeasibility tests, which require a little additional computation, not only for primal linear programs $\mathrm{P}(K, \boldsymbol{F})$ but for their dual programs. To apply the primal simplex method to the primal linear program $\mathrm{P}(K, \boldsymbol{F})$, we need to convert each of the less-than inequalities in $\mathrm{P}(K, \boldsymbol{F})$ to equality form by adding a nonnegative *slack* variable. The resulting problem is called a *standard form* linear program. The transformation of $\mathrm{P}(K, \boldsymbol{F})$ into the standard form increases the number of variables considerably. On the other hand, the dual program of $\mathrm{P}(K, \boldsymbol{F})$ is already in the standard form without additional variables as $\mathrm{D}(K, \boldsymbol{F})$ of Section 4.2 shows, and thus, easier to deal with than $\mathrm{P}(K, \boldsymbol{F})$.

4.1 Feasibility/infeasibility tests using primal problems.

We will consider the problem $\mathrm{P}(K, \boldsymbol{F})$ with $\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{\leq}(K)$, $K \subseteq M$, any $b_j \in R$ $(j \in K)$ and any $\boldsymbol{d} \in R^n$. We assume that the problem $\mathrm{P}(K, \boldsymbol{F})$ is feasible, and that a basic feasible solution

$(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K)$ satisfying the properties below is available.

$$\begin{array}{ll} \bar{\beta}_j - \langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle < \omega_j(\boldsymbol{a}) & (\boldsymbol{a} \in N_j,\ j \in K), \\ \bar{\beta}_j - \langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle = \omega_j(\boldsymbol{a}) & (\boldsymbol{a} \in B_j,\ j \in K), \\ F_j \subseteq B_j \subseteq \mathcal{A}_j,\ N_j \subset \mathcal{A}_j,\ B_j \cap N_j = \emptyset,\ \#B_j + \#N_j = \#\mathcal{A}_j & (j \in K), \\ \sum_{j \in K} \#B_j = n + \#K. & \end{array} \tag{8}$$

Note that $\mathrm{P}(K, \boldsymbol{F})$ has $(\sum_{j \in K} \#\mathcal{A}_j)$ linear equality/inequality constraints, and $(n + \#K)$ variables; $\boldsymbol{\alpha} \in R^n$ and β_j $(j \in K)$. The set B_j indicates active constraints among the jth-level constraints $\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a})$, $\boldsymbol{a} \in \mathcal{A}_j$. Then, we see that $\sum_{j \in K} \#B_j$ is equal to $(n + \#K)$; the number of variables. Here, we introduce a feasibility test for $P(K, \boldsymbol{F})$ using the set B_j $(j \in K)$.

Lemma 4.1. *(Primal feasibility test)*
Let $\bar{K} \subseteq K \subseteq M$. For the linear program $P(K, \boldsymbol{F})$ with $\boldsymbol{F} \in \mathcal{F}^{\leq}(K)$, define B_j $(\forall j \in K)$ at some feasible basic solution $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K)$ as in (8). If $\boldsymbol{F}' = (F'_j : j \in \bar{K}) \in \mathcal{F}^{\leq}(\bar{K})$ satisfies $F'_j \subseteq B_j$ $(\forall j \in \bar{K})$, then the problem $P(\bar{K}, \boldsymbol{F}')$ is feasible.

Proof: The partial vector $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in \bar{K})$ of $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K)$ satisfies $\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a})$ $(\boldsymbol{a} \in F'_j,\ j \in \bar{K})$. Therefore, $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in \bar{K})$ is a feasible solution of $\mathrm{P}(\bar{K}, \boldsymbol{F}')$, and this lemma follows. ∎

Starting from the basic feasible solution $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K)$, our enumeration method applies the simplex method to the problem $\mathrm{P}(K, \boldsymbol{F})$ with $\boldsymbol{F} \in \mathcal{F}^{\leq}(K)$. At each iteration, we can apply the primal feasibility test of Lemma 4.1, which might detect feasibility of some other problems $\mathrm{P}(K, \boldsymbol{F}')$ with $\boldsymbol{F}' \in \mathcal{F}^{\leq}(K)$. Now, we will provide an optimality test related to the feasibility test of Lemma 4.1. By utilizing the optimality test effectively on the enumeration tree, we can reduce the number of linear programs to be solved.

Lemma 4.2. *(Primal optimality test)*
Let $K \subseteq M$. Assuming that the linear program $P(K, \boldsymbol{F})$ with $\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{\leq}(K)$ has an optimal solution $(\boldsymbol{\alpha}^*, \beta_j^* : j \in K)$, define B_j $(\forall j \in K)$ at the optimal solution as in (8). If $\boldsymbol{F}' = (F'_j : j \in K) \in \mathcal{F}^{\leq}(K)$ satisfies $F_j \subseteq F'_j \subseteq B_j$ $(\forall j \in K)$, then $(\boldsymbol{\alpha}^*, \beta_j^* : j \in K)$ is also optimal solution for the problem $P(K, \boldsymbol{F}')$.

Proof: Lemma 4.1 ensures that $(\boldsymbol{\alpha}^*, \beta_j^* : j \in K)$ is a feasible solution for $\mathrm{P}(K, \boldsymbol{F}')$. Since this inclusion $F_j \subseteq F'_j$ for $\forall j \in K$ means that the feasible region of $\mathrm{P}(K, \boldsymbol{F}')$ is smaller than that of $\mathrm{P}(K, \boldsymbol{F})$, the solution $(\boldsymbol{\alpha}^*, \beta_j^* : j \in K)$ is optimal not only for $\mathrm{P}(K, \boldsymbol{F})$ but for $\mathrm{P}(K, \boldsymbol{F}')$. ∎

Next, we will provide a device to check the primal infeasibility for a node $\mathrm{P}(K, \widetilde{\boldsymbol{F}})$ with $\widetilde{\boldsymbol{F}} \in \mathcal{F}^{\leq}(K)$, using solution information already obtained. For some feasible problem $\mathrm{P}(K, \boldsymbol{F})$ with $\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{\leq}(K)$, take $\boldsymbol{G} = (G_j : j \in K) \in \mathcal{F}^{\leq}(K)$ whose components satisfy $G_j \supseteq F_j$ $(\forall j \in K)$. Using randomly generated numbers $x_j(\boldsymbol{a}) > 0$ $(\forall \boldsymbol{a} \in G_j,\ \forall j \in K)$, define coefficients of the objective function in $\mathrm{P}(K, \boldsymbol{F})$ as

$$b_j = \sum_{\bar{\boldsymbol{a}} \in G_j} x_j(\boldsymbol{a}) \ \ (j \in K) \ \text{ and } \boldsymbol{d} = \sum_{j \in K} \sum_{\bar{\boldsymbol{a}} \in G_j} \bar{\boldsymbol{a}}\, x_j(\boldsymbol{a}). \tag{9}$$

Then, the objective function can be denoted as

$$\sum_{j \in K} b_j \beta_j - \langle \boldsymbol{d}, \boldsymbol{\alpha} \rangle = \sum_{j \in K} \sum_{\bar{\boldsymbol{a}} \in G_j} x_j(\boldsymbol{a}) \left(\beta_j - \langle \bar{\boldsymbol{a}}, \boldsymbol{\alpha} \rangle \right). \tag{10}$$

Since the objective value is bounded from above as follows; $\sum_{j \in K} b_j \beta_j - \langle \boldsymbol{d}, \boldsymbol{\alpha} \rangle \leq \sum_{j \in K} \sum_{\bar{\boldsymbol{a}} \in G_j} x_j(\boldsymbol{a})\, \omega_j(\bar{\boldsymbol{a}})$, the primal simplex method results in an optimal solution $(\boldsymbol{\alpha}^*, \beta_j^* : j \in K)$ of the problem $\mathrm{P}(K, \boldsymbol{F})$.

Lemma 4.3. *(Primal infeasibility test)*
Let $\widetilde{\boldsymbol{F}} = (\widetilde{F}_j : j \in K) \in \mathcal{F}^{\subseteq}(K)$, *where* $\widetilde{F}_j \supseteq G_j$ $(j \in K)$. *If the optimal solution* $(\boldsymbol{\alpha}^*, \beta_j^* : j \in K)$ *of* $P(K, \boldsymbol{F})$ *satisfies*

$$\sum_{j \in K} b_j \beta_j^* - \langle \boldsymbol{d}, \boldsymbol{\alpha}^* \rangle < \sum_{j \in K} \sum_{\bar{\boldsymbol{a}} \in G_j} x_j(\boldsymbol{a})\ \omega_j(\bar{\boldsymbol{a}}), \tag{11}$$

then the problem $P(K, \widetilde{\boldsymbol{F}})$ *is infeasible.*

Proof: Taking notice that the problem $\mathrm{P}(K, \boldsymbol{F})$ maximizes the objective function of (10), the strict inequality of (11) means infeasibility of $\mathrm{P}(K, \boldsymbol{G})$, and also infeasibility of $\mathrm{P}(K, \widetilde{\boldsymbol{F}})$. ∎

If the optimal value of $\mathrm{P}(K, \boldsymbol{F})$ satisfies (11) for some set $\boldsymbol{G} = (G_j : j \in K) \in \mathcal{F}^{\subseteq}(K)$, we detect the infeasibility of $\mathrm{P}(K, \widetilde{\boldsymbol{F}})$ for $\forall \widetilde{\boldsymbol{F}}$ such that $\widetilde{F}_j \supseteq G_j$ $(j \in K)$, without solving any linear program.

4.2 Feasibility/infeasibility tests using dual problems.

Corresponding to each $\mathrm{P}(K, \boldsymbol{F})$ with some $\boldsymbol{F} \in \mathcal{F}^{\subseteq}(K)$ and $K \subseteq M$, we consider its dual linear program $\mathrm{D}(K, \boldsymbol{F})$.

$$\begin{array}{lll}
\mathrm{D}(K, \boldsymbol{F}): \quad \text{minimize} & \displaystyle\sum_{j \in K} \sum_{\boldsymbol{a} \in \mathcal{A}_j} \omega_j(\boldsymbol{a}) x_j(\boldsymbol{a}) & \\
\text{subject to} & \displaystyle\sum_{j \in K} \sum_{\boldsymbol{a} \in \mathcal{A}_j} \boldsymbol{a}\, x_j(\boldsymbol{a}) = \boldsymbol{d}, & \\
& \displaystyle\sum_{\boldsymbol{a} \in \mathcal{A}_j} x_j(\boldsymbol{a}) = b_j & (j \in K), \\
& x_j(\boldsymbol{a}) \geq 0 & (\boldsymbol{a} \in \mathcal{A}_j \backslash F_j,\ j \in K), \\
& -\infty < x_j(\boldsymbol{a}) < +\infty & (\boldsymbol{a} \in F_j,\ j \in K).
\end{array}$$

It should be noted that the inequality constraints $\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a})$ $(\boldsymbol{a} \in \mathcal{A}_j \backslash F_j,\ j \in K)$ and the equality constraints $\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a})$ $(\boldsymbol{a} \in F_j,\ j \in K)$ in the primal subproblem $\mathrm{P}(K, \boldsymbol{F})$ are corresponding to the nonnegative variables $x_j(\boldsymbol{a}) \geq 0$ $(\boldsymbol{a} \in \mathcal{A}_j \backslash F_j,\ j \in K)$ and the free variables $-\infty < x_j(\boldsymbol{a}) < +\infty$ $(\boldsymbol{a} \in F_j,\ j \in K)$ in the dual subproblem $\mathrm{D}(K, \boldsymbol{F})$, respectively. $\mathrm{D}(K, \boldsymbol{F})$ includes no inequality constraints except for nonnegative constraints such that $x_j(\boldsymbol{a}) \geq 0$ $(\boldsymbol{a} \in \mathcal{A}_j \backslash F_j,\ j \in K)$. So no additional slack variables are required for $\mathrm{D}(K, \boldsymbol{F})$ by the primal simplex method. $\mathrm{D}(K, \boldsymbol{F})$ has $\sum_{j \in K} \#\mathcal{A}_j$ variables and $(n + \#K)$ linear constraints; numerical results show that these numbers are much less in dual problems $\mathrm{D}(K, \boldsymbol{F})$ than in primal $\mathrm{P}(K, \boldsymbol{F})$ of the standard form. Therefore, it is preferable to deal with $\mathrm{D}(K, \boldsymbol{F})$ instead of $\mathrm{P}(K, \boldsymbol{F})$.

For any $\boldsymbol{F} \in \mathcal{F}^{\subseteq}(K)$ and $K \subseteq M$, we now describe a logical test to check whether $\mathrm{P}(K, \boldsymbol{F})$ is feasible or not, by applying the primal simplex method to $\mathrm{D}(K, \boldsymbol{F})$. We assume for the time being that $b_j \in R$ $(j \in K)$ and $\boldsymbol{d} \in R^n$ are chosen so that the problem $\mathrm{D}(\boldsymbol{F}, K)$ is feasible, and that a basic feasible solution $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K)$ of $\mathrm{D}(\boldsymbol{F}, K)$ is available. We will discuss how to construct such a basic solution later. For every $j \in K$, let

$$\left.\begin{array}{rcl}
B_j & = & \{\bar{\boldsymbol{a}} \in \mathcal{A}_j : \bar{x}_j(\bar{\boldsymbol{a}}) \ \text{ is a basic variable at } \bar{\boldsymbol{x}}\}, \\
N_j & = & \{\boldsymbol{a} \in \mathcal{A}_j : \bar{x}_j(\boldsymbol{a}) \ \text{ is a nonbasic variable at } \bar{\boldsymbol{x}}\}.
\end{array}\right\} \tag{12}$$

Then we can rewrite the problem $\mathrm{D}(\boldsymbol{F}, K)$ as

$$\left.\begin{array}{lll}
\text{minimize} & \displaystyle\sum_{j \in K} \sum_{\boldsymbol{a} \in N_j} \bar{\omega}_j(\boldsymbol{a})\ x_j(\boldsymbol{a}) + \bar{\zeta}_0 & \\
\text{subject to} & \displaystyle x_i(\bar{\boldsymbol{a}}) = \bar{x}_i(\bar{\boldsymbol{a}}) + \sum_{j \in K} \sum_{\boldsymbol{a} \in N_j} \bar{g}_{ij}(\bar{\boldsymbol{a}}, \boldsymbol{a})\ x_j(\boldsymbol{a}) & (\bar{\boldsymbol{a}} \in B_i,\ i \in K), \\
& x_j(\boldsymbol{a}) \geq 0 & (\boldsymbol{a} \in \mathcal{A}_j \backslash F_j,\ j \in K), \\
& -\infty < x_j(\boldsymbol{a}) < +\infty & (\boldsymbol{a} \in F_j,\ j \in K),
\end{array}\right\} \tag{13}$$

where the coefficients

$$\left.\begin{array}{ll} \bar{\omega}_j(\boldsymbol{a}) \in R & (\boldsymbol{a} \in N_j,\ j \in K), \\ \bar{\zeta}_0 = \sum_{i \in K} \sum_{\bar{\boldsymbol{a}} \in B_i} \omega_i(\bar{\boldsymbol{a}})\ \bar{x}_i(\bar{\boldsymbol{a}}) & \in R, \\ \bar{g}_{ij}(\bar{\boldsymbol{a}}, \boldsymbol{a}) \in R & (\bar{\boldsymbol{a}} \in B_i,\ i \in K,\ \boldsymbol{a} \in N_j,\ j \in K), \end{array}\right\} \tag{14}$$

can be obtained from the simplex tableau or the dictionary with the basic feasible solution $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K)$. It follows from Condition 2.2 that $\bar{\omega}_j(\boldsymbol{a}) \neq 0$ for $\forall \boldsymbol{a} \in N_j$ and $\forall j \in K$. We can reach the dual infeasibility test for dual programs $\mathrm{D}(K, \boldsymbol{F})$ as well as the primal infeasibility test. Also, the dual optimality test is same as the primal test of Lemma 4.2. Now, we will present an infeasibility test applicable to any $\hat{\boldsymbol{a}} \in N_j$ with $\forall j \in K$, which examines whether some linear problem $\mathrm{P}(K, \boldsymbol{F})$ becomes infeasible by changing the nonbasic variable $x_j(\hat{\boldsymbol{a}})$ into basic one.

Lemma 4.4. *(Dual infeasibility test)*
Let $\hat{\boldsymbol{a}} \in N_j$ with some $j \in K$.

(a) Assume that $\bar{\omega}_j(\hat{\boldsymbol{a}}) < 0$. Let $\boldsymbol{F}' = (F'_j : j \in K)$ be $F'_i = \{\bar{\boldsymbol{a}} \in B_i : \bar{g}_{ij}(\bar{\boldsymbol{a}}, \hat{\boldsymbol{a}}) < 0\}$ $(i \in K)$. If $\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{\leq}(K)$ satisfies $F'_i \subseteq F_i$ for $\forall i \in K$, then the problem $P(K, \boldsymbol{F})$ is infeasible.

(b) Assume that $\bar{\omega}_j(\hat{\boldsymbol{a}}) > 0$. Let $\boldsymbol{F}' = (F'_j : j \in K)$ be $F'_i = \{\bar{\boldsymbol{a}} \in B_i : \bar{g}_{ij}(\bar{\boldsymbol{a}}, \hat{\boldsymbol{a}}) > 0\}$ $(i \in K,\ i \neq j)$ and $F'_j = \{\boldsymbol{a} \in B_j : \bar{g}_{jj}(\bar{\boldsymbol{a}}, \hat{\boldsymbol{a}}) > 0\} \cup \{\hat{\boldsymbol{a}}\}$. If $\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{\leq}(K)$ satisfies $F'_i \subseteq F_i$ for $\forall i \in K$, then the problem $P(K, \boldsymbol{F})$ is infeasible.

Proof: In (a), in making $x_j(\hat{\boldsymbol{a}})$ basic variable from nonbasic variable, we can increase the variable $x_j(\hat{\boldsymbol{a}})$ to $+\infty$ while keeping the feasibility of $(x_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j, j \in K)$. This means that the problem $\mathrm{D}(K, \boldsymbol{F}')$ is unbounded and also the corresponding primal problem $\mathrm{P}(K, \boldsymbol{F})$ is infeasible. The infeasibility of the case (b) occurs only when the jth element F_j of $\boldsymbol{F}$ includes $\hat{\boldsymbol{a}}$, that is, when $x_j(\hat{\boldsymbol{a}})$ is a free variable which can take negative value. If the condition of (b) is satisfied for $\mathrm{D}(K, \boldsymbol{F})$, $x_j(\hat{\boldsymbol{a}})$ can be decreased to $-\infty$ and $\mathrm{D}(K, \boldsymbol{F})$ is found unbounded. ∎

4.3 Updating basic feasible solutions.

Throughout this section, we assume that $K_0, K_1, \ldots, K_\ell$ are taken as in (7), and discuss how to update the basis information when we proceed from the pth level to the $(p+1)$st level. Let $\bar{\boldsymbol{F}} = \{\bar{F}_1, \ldots, \bar{F}_p\} \in \mathcal{F}^{=\&\mathrm{f}}(K_p)$ and $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_1, \bar{\beta}_2, \ldots, \bar{\beta}_p)$ be an optimal solution of $\mathrm{P}(K_p, \bar{\boldsymbol{F}})$ satisfying the relations (8). Next, for the $(p+1)$st set of inequalities $\beta_{p+1} - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_{p+1}(\boldsymbol{a})$ $(\forall \boldsymbol{a} \in \mathcal{A}_{p+1})$, define

$$\begin{aligned} \bar{\beta}_{p+1} &= \min\{\langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle + \omega_{p+1}(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_{p+1}\} \\ &= \langle \tilde{\boldsymbol{a}}, \bar{\boldsymbol{\alpha}} \rangle + \omega_{p+1}(\tilde{\boldsymbol{a}}) \text{ for some } \tilde{\boldsymbol{a}} \in \mathcal{A}_{p+1}, \\ B_{p+1} &= \{\tilde{\boldsymbol{a}}\},\quad N_{p+1} = \{\boldsymbol{a} \in \mathcal{A}_{p+1} : \boldsymbol{a} \neq \tilde{\boldsymbol{a}}\}. \end{aligned}$$

Using $\bar{\boldsymbol{F}} = \{\bar{F}_1, \ldots, \bar{F}_p\} \in \mathcal{F}^{=\&\mathrm{f}}(K_p)$, define $\boldsymbol{F}$ as

$$\boldsymbol{F} = (\bar{F}_1, \ldots, \bar{F}_p, \{\emptyset\}) \in \mathcal{F}^{\leq}(K_{p+1}), \text{ or} \tag{15a}$$

$$\boldsymbol{F} = (\bar{F}_1, \ldots, \bar{F}_p, \{\tilde{\boldsymbol{a}}\}) \in \mathcal{F}^{\leq}(K_{p+1}), \tag{15b}$$

and let $\boldsymbol{d}$ and b_j $(j = 1, \ldots, p+1)$ in $\mathrm{P}(K_{p+1}, \boldsymbol{F})$ (or its dual $\mathrm{D}(K_{p+1}, \boldsymbol{F})$) be

$$\bar{x}_j(\boldsymbol{a}) = \begin{cases} \text{a positive number generated randomly} & (\text{ if } \boldsymbol{a} \in B_j,\ j = 1, 2, \ldots, p+1), \\ 0 & (\text{ if } \boldsymbol{a} \in N_j,\ j = 1, 2, \ldots, p+1), \end{cases}$$

$$\boldsymbol{d} = \sum_{j=1}^{p+1} \sum_{\boldsymbol{a} \in B_j} \boldsymbol{a}\ \bar{x}_j(\boldsymbol{a}),\quad b_j = \sum_{\boldsymbol{a} \in B_j} \bar{x}_j(\boldsymbol{a})\ (j = 1, 2, \ldots, p+1).$$

Then $(\bar{\alpha}, \bar{\beta}_1, \bar{\beta}_2, \ldots, \bar{\beta}_p, \bar{\beta}_{p+1})$ turns out to be an optimal solution for $\mathrm{P}(K_{p+1}, \boldsymbol{F})$, and $(\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j = 1, 2, \ldots, p+1)$ for the dual $\mathrm{D}(K_{p+1}, \boldsymbol{F})$. Note that the above construction of these vectors $\boldsymbol{d}$ and b_j $(j = 1, 2, \ldots, p+1)$ has no influence on the feasibility of $\mathrm{P}(K_{p+1}, \boldsymbol{F})$ and the unboundedness of $\mathrm{D}(K_{p+1}, \boldsymbol{F})$.

The new basic variables of $\mathrm{P}(K_{p+1}, \boldsymbol{F})$ include only one additional variable, besides the old ones of $\mathrm{P}(K_p, \bar{\boldsymbol{F}})$; in the primal problem $\mathrm{P}(K_{p+1}, \boldsymbol{F})$, the additional basic variable corresponds to β_{p+1} and in its dual $\mathrm{D}(K_{p+1}, \boldsymbol{F})$, corresponds to $x_j(\tilde{\boldsymbol{a}})$. Therefore, we can easily generate an optimal dictionary of $\mathrm{P}(K_{p+1}, \boldsymbol{F})$ (or $\mathrm{D}(K_{p+1}, \boldsymbol{F})$), based on the previous optimal dictionary of $\mathrm{P}(K_p, \bar{\boldsymbol{F}})$ (or $\mathrm{D}(K_p, \bar{\boldsymbol{F}})$). For example, consider a dictionary of $\mathrm{D}(K_{p+1}, \boldsymbol{F})$, that is, the form (13). For its additional reduced cost vector $(\bar{\omega}_{p+1}(\boldsymbol{a}) : \boldsymbol{a} \in N_{p+1})$, we see that $\bar{\omega}_{p+1}(\boldsymbol{a}) = \langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle + \omega_{p+1}(\boldsymbol{a}) - \bar{\beta}_{p+1}$ and $\bar{\omega}_{p+1}(\boldsymbol{a}) > 0$ for $\forall \boldsymbol{a} \in N_{p+1}$.

After obtaining the optimal dictionary of $\mathrm{P}(K_{p+1}, \boldsymbol{F})$ (or $\mathrm{D}(K_{p+1}, \boldsymbol{F})$) for $\boldsymbol{F}$ of (15a), choose any $\hat{\boldsymbol{a}} \in \mathcal{A}_{p+1}$ and check whether the problem $\mathrm{P}(K_{p+1}, \hat{\boldsymbol{F}})$ with $\hat{\boldsymbol{F}} = (\bar{F}_1, \ldots, \bar{F}_p, \{\hat{\boldsymbol{a}}\}) \in \mathcal{F}^{\leq}(K_{p+1})$ is feasible or not, by applying infeasibility tests of Lemma 4.3 or Lemma 4.4 to the dictionary of $\mathrm{P}(K_{p+1}, \boldsymbol{F})$ (or $\mathrm{D}(K_{p+1}, \boldsymbol{F})$). Note that if $\hat{\boldsymbol{a}} \in \mathcal{A}_{p+1}$ is equal to $\tilde{\boldsymbol{a}}$, then the optimal dictionaries of $\mathrm{P}(K_{p+1}, \hat{\boldsymbol{F}})$ and $\mathrm{D}(K_{p+1}, \hat{\boldsymbol{F}})$ are already obtained. In the following discussion, we assume that $\hat{\boldsymbol{a}} \neq \tilde{\boldsymbol{a}}$.

Primal Case: We reset the objective function $\sum_{j=1}^{p+1} b_j \beta_j - \langle \boldsymbol{d}, \boldsymbol{\alpha} \rangle$ of $\mathrm{P}(K_{p+1}, \boldsymbol{F})$ by in (9), taking $K = K_{p+1}$, $G_j = \bar{F}_j$ $(j = 1, \ldots, p)$, $x_j(\boldsymbol{a}) = \bar{x}_j(\boldsymbol{a})$ $(\forall \boldsymbol{a} \in G_j,\ j = 1, \ldots, p)$ and $G_{p+1} = \{\hat{\boldsymbol{a}}\}$, and setting $x_{p+1}(\hat{\boldsymbol{a}}) > 0$ randomly. From the feasible solution $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_1, \ldots, \bar{\beta}_{p+1})$, we start the pivoting procedure and in the end, compare the optimal value of $\mathrm{P}(K_{p+1}, \boldsymbol{F})$ with $\displaystyle\sum_{j=1}^{p+1} \sum_{\bar{\boldsymbol{a}} \in G_j} x_j(\bar{\boldsymbol{a}})\ \omega_j(\bar{\boldsymbol{a}})$, as the primal infeasibility test of Lemma 4.3 shows.

Dual Case: Since $\hat{\boldsymbol{a}} \in N_{p+1}$ and $\bar{\omega}_{p+1}(\hat{\boldsymbol{a}}) > 0$ holds, we apply the dual infeasibility test (b) of Lemma 4.4 to the optimal dictionary (13) of $\mathrm{D}(K_{p+1}, \boldsymbol{F})$. For $(F'_1, \ldots, F'_{p+1})$ defined by Lemma 4.4 (b), if $\hat{\boldsymbol{F}} = (\bar{F}_1, \ldots, \bar{F}_p, \{\hat{\boldsymbol{a}}\})$ satisfies $F'_i \subseteq \bar{F}_i$, $(i = 1, \ldots, p)$, then we find that $\mathrm{D}(K_{p+1}, \hat{\boldsymbol{F}})$ is unbounded, i.e., $\mathrm{P}(K_{p+1}, \hat{\boldsymbol{F}})$ is infeasible. Otherwise, apply the pivoting procedure for $\mathrm{D}(K_{p+1}, \hat{\boldsymbol{F}})$ while checking the feasibility of $\mathrm{P}(K_{p+1}, \hat{\boldsymbol{F}})$ by Lemma 4.4 (a).

If $\mathrm{P}(K_{p+1}, \hat{\boldsymbol{F}})$ is feasible, we add another element $\boldsymbol{a} \in \mathcal{A}_{p+1}$ to the $(p+1)$st element of $\hat{\boldsymbol{F}}$, apply infeasibility checks corresponding to the new set $\hat{\boldsymbol{F}}$, and continue this procedure until $\mathrm{P}(K_{p+1}, \hat{\boldsymbol{F}})$ becomes infeasible or $\hat{\boldsymbol{F}} \in \mathcal{F}^{=\&\mathrm{f}}(K_{p+1})$.

5 Implementation

In this section, to simplify our discussion, we focus on a fully mixed polynomial system (1), and also, the dual reformulation for linear systems induced from Problem 2.1'. We provide two algorithms for Problem 2.1'; one is a sequential algorithm and the other is a parallel algorithm on a client-server based parallel computing system.

5.1 Serial and Parallel Enumeration Algorithms

Every fully mixed system has $m = n$, $M = \{1, 2, \ldots, n\}$ and $(s_1, \ldots, s_n) = (1, \ldots, 1)$. We give some order among the elements of $\mathcal{A}_j$ $(j \in M)$ and denote them as $\boldsymbol{a}_j(1), \boldsymbol{a}_j(2), \ldots, \boldsymbol{a}_j(m_j)$, where $m_j = \sharp \mathcal{A}_j$, $(j \in M)$. We consider all possible distinct pairs $\{\boldsymbol{a}_j(p), \boldsymbol{a}_j(q)\}$ of $\mathcal{A}_j$ with $1 \leq p < q \leq m_j$ and arrange them in the lexicographical order, i.e.,

$$L(\mathcal{A}_j) \equiv \{\{\boldsymbol{a}_j(1), \boldsymbol{a}_j(2)\}, \{\boldsymbol{a}_j(1), \boldsymbol{a}_j(3)\}, \ldots, \{\boldsymbol{a}_j(m_j - 1), \boldsymbol{a}_j(m_j)\}\},$$

where $\boldsymbol{a}_j(1), \boldsymbol{a}_j(2), \ldots, \boldsymbol{a}_j(m_j) \in \mathcal{A}_j$. For every $F_j = \{\boldsymbol{a}_j(p), \boldsymbol{a}_j(q)\}$ in the list $L(\mathcal{A}_j)$, we define

$$\mathrm{succ}(F_j; L(\mathcal{A}_j)) = \begin{cases} \emptyset & (\text{if } F_j \text{ is the last element in the list } L(\mathcal{A}_j)), \\ \text{the element succeeding to } F_j \text{ in the list } L(\mathcal{A}_j) & (\text{otherwise}), \end{cases}$$

and let $\mathrm{succ}(\emptyset; L(\mathcal{A}_j)) =$ the first element in the list $L(\mathcal{A}_j)$. Also, for description of algorithms, we define K_p and $\boldsymbol{F}_p$ as follows; $K_p = \{1, 2, \ldots, p\}$ for $\forall p \in M$, and $\boldsymbol{F}_p = \{F_1, F_2, \ldots, F_p\} \in \mathcal{F}^{\leq}(K_p)$.

Algorithm 5.1. (Serial Enumeration Algorithm)

Step 0: Let $\widetilde{\mathcal{A}_1} = \mathcal{A}_1$ and $p = 1$.

Step 1: Solve $\mathrm{D}(K_p, \boldsymbol{F}_p)$ with $F_p = \{\boldsymbol{a}\}$ for $\forall \boldsymbol{a} \in \widetilde{\mathcal{A}_p}$. For some $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}_p}$, if $\mathrm{D}(K_p, \boldsymbol{F}_p)$ with $F_p = \{\bar{\boldsymbol{a}}\}$ is unbounded, delete such $\bar{\boldsymbol{a}}$ from $\widetilde{\mathcal{A}_p}$, i.e., $\widetilde{\mathcal{A}_p} = \widetilde{\mathcal{A}_p} \backslash \{\bar{\boldsymbol{a}}\}$. Let $F_p = \emptyset$ and go to Step 2.

Step 2: If $p = 0$ then terminate. Otherwise, let

$$F_j = \begin{cases} F_j & \text{if } 1 \leq j \leq p-1, \\ \mathrm{succ}(F_p, L(\widetilde{\mathcal{A}_p})) & \text{if } j = p. \end{cases}$$

Step 3: If $F_p = \emptyset$, then let $p = p - 1$ and go to Step 2. Otherwise, go to Step 4.

Step 4: Solve $\mathrm{D}(K_p, \boldsymbol{F}_p)$ to compute a basic optimal solution $\boldsymbol{x} = (x_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K_p)$ or detect its unboundedness. If $\mathrm{D}(K_p, \boldsymbol{F}_p)$ is unbounded, go to Step 2. Otherwise go to Step 5.

Step 5: If $p = n$, then output the optimal solution of $\mathrm{P}(K_p, \boldsymbol{F}_p)$ and go to Step 2. Else, go to Step 6.

Step 6: Letting $F_{p+1} = \emptyset$, obtain the optimal dictionary of $\mathrm{D}(K_{p+1}, \boldsymbol{F}_{p+1})$ as we have described in the previous section. Let $p = p + 1$ and $\widetilde{\mathcal{A}_p} = \mathcal{A}_p$. Select some or all of the nonbasic variables $x_p(\bar{\boldsymbol{a}})$ with $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}_p}$, and check whether pivoting each selected nonbasic variable $x_p(\bar{\boldsymbol{a}})$ into new basic variables leads to the unbounded objective value; if such the case happens, let $\widetilde{\mathcal{A}_p} = \widetilde{\mathcal{A}_p} \backslash \{\bar{\boldsymbol{a}}\}$. Go to Step 1.

Steps 1 and 6 try to detect some elements $\bar{\boldsymbol{a}} \in \mathcal{A}_p$ such that linear programs $D(K_p, \boldsymbol{F}_p)$ including $\bar{\boldsymbol{a}} \in \mathcal{A}_p$ in F_p are unbounded. By eliminating $\bar{\boldsymbol{a}} \in \mathcal{A}_p$ from $\widetilde{\mathcal{A}_p}$, the number of elements in the list $L(\widetilde{\mathcal{A}_p})$ decreases considerably and thus, the number of linear programs to be solved at Step 4 decreases. At Step 6, to find elements $\bar{\boldsymbol{a}} \in \mathcal{A}_p$ which lead the problem $P(K_p, \boldsymbol{F}_p)$ to infeasibility, we utilize Lemma 4.4 with no additional calculation. At Step 1 we make, moreover, unboundedness-checks for the remaining $\bar{\boldsymbol{a}} \in \mathcal{A}_p$ in $\widetilde{\mathcal{A}_p}$. Although we need to solve $\#\widetilde{\mathcal{A}_p}$ additional linear programs at Step 1, this operation leads to a significant reduction in the number of linear programs to be solved, consequently.

In the rest of this section, we will show a modified algorithm of Algorithm 5.1 for parallel computing. Now suppose that we have a client-server based computer system consisting of N processors. The paralleled Algorithm 5.2 requires the enumeration tree for the first p^* levels, where p^* satisfies $N < \#\mathcal{F}^{=}(K_{p^*}) = \frac{1}{2}\Pi_{i=1}^{p^*} m_i(m_i - 1)$. In this scheme, $\#\mathcal{F}^{=}(K_{p^*})$ nodes of the p^*th level are each assigned to individual processors, and from each node regarded as a root node, an enumeration subtree is constructed using the same strategy depicted in Algorithm 5.1. Each processor can process this part independently without any communication among server machines.

Algorithm 5.2. (Parallel Enumeration Algorithm)

Step 0: Choose an initial depth p^* such that $N < \#\mathcal{F}^{=}(K_{p^*})$.

Table 1: Computational CPU Time

problem	serial Algorithm 5.1		MVLP		Li&Li	
	CPU time	Memory	CPU time	Memory	CPU time	Memory
cyclic8	1.8s	2520K	41.1s	21M	1.8s	1040K
cyclic9	20.4s	2688K	7m 03.7s	21M	11.7s	1072K
cyclic10	3m 07.8s	2744K	1h 00m 23.0s	21M	1m 42.9s	1096K
cyclic11	36m 40.3s	2856K	9h 43m 54.0s	21M	18m 06.0s	1136K
eco9	7.7s	2512K	25.0s	21M	9.2s	2352K
eco10	48.3s	2672K	2m 04.3s	21M	53.4s	2352K
eco11	5m 03.9s	2792K	10m 12.7s	21M	6m 28.3s	2408K
eco12	31m 27.3s	2936K	1h 06m 23.3s	21M	41m 34.5s	2408K

Step 1: Assign $\boldsymbol{F} \in \mathcal{F}^{=}(K_{p^*})$ to some server machine with an idle processor, and delete $\boldsymbol{F}$ from $\mathcal{F}^{=}(K_{p^*})$. The assignment continues until no idle processor exists or $\mathcal{F}^{=}(K_{p^*})$ becomes empty.

Step 2: On every server machine given some $\widetilde{\boldsymbol{F}} = \{\widetilde{F_1}, \widetilde{F_2}, \ldots, \widetilde{F_{p^*}}\} \in \mathcal{F}^{=}(K_{p^*})$ by the client machine, solve subproblems $\mathrm{D}(K_p, \boldsymbol{F}_p)$ for $p > p^*$ by setting

$$F_j = \begin{cases} \widetilde{F_j} & \text{if } 1 \le j \le p^*, \\ F_j & \text{if } p^* + 1 \le j \le p-1, \\ \mathrm{succ}(F_p, L(\widetilde{\mathcal{A}_p})) & \text{if } j = p, \end{cases}$$

according to Algorithm 5.1. If the termination criteria of Algorithm 5.1 is satisfied on some server machine, return solutions of $\mathrm{D}(K_n, \boldsymbol{F}_n)$ with $\boldsymbol{F}_n \in \mathcal{F}^{=\&\mathrm{f}}(K_n)$ such that $F_j = \widetilde{F_j}$ $(j = 1, \ldots, p^*)$, to the client machine. If $\mathcal{F}^{=}(K_{p^*})$ is empty, go to Step 3. Otherwise, go to Step 1.

Step 3: After Algorithm 5.1 terminates on all active processor, collect all solutions from each server machine and output them. They are solutions of Problem 2.1'.

5.2 Numerical Results

We chose two kinds of n-dimensional benchmark systems for numerical experiments; cyclic n-roots problem [2] and n-dimensional economics problem [8]. Corresponding to these benchmark polynomial systems, we construct Problem 2.1' with elements $\omega_j(\boldsymbol{a})$ $(\forall \boldsymbol{a} \in \mathcal{A}_j, \forall j \in M)$ randomly generated in the interval (0, 50).

In order to compare our algorithm with some existing methods for mixed cell computation, we solved our test problems using standard serial codes based on Algorithm 5.1. The program was coded in C++ language and was ran on a DEC Alpha Workstation (CPU 21164 600MHz with 1GB memory). Table 1 shows the computational CPU time for cyclic n-roots problems (abbreviated by cyclic-n) and n-dimensional economics problems (abbreviated by eco-n), comparing our method with the existing software packages such as MVLP [3] and Li&Li [7]. Note that our method extremely outperforms MVLP in terms of CPU time and also, required memory storage. There is, however, no significant difference between our serial Algorithm 5.1 and Li&Li algorithm in the used memory and CPU time. In n-dimensional economic problems, our method enumerated solutions of Problem 2.1' faster than Li&Li, while ours was defeated with respect to cyclic n-roots problems. Our algorithm requires more memory storage than Li&Li's, since ours stores the data for explicit representations of the basis inverses, which obtained by solving linear programs at Steps 1 and 4 of Algorithm 5.1. By utilizing the stored inverse matrices for solving linear programs $D(K_p, \boldsymbol{F}_p)$, we reduced the computation related to inverse operation of matrices.

Table 2: Computational Real Time on Parallel Computing System

	Cyclic-n Problems				Eco-n Problems		
N	cyclic11	cyclic12	cyclic13	cyclic14	eco12	eco13	eco14
1	27m 27s	4h 00m 03s			22m 59s	2h 19m 59s	
2	13m 52s	2h 00m 14s			11m 26s	1h 09m 60s	
4	6m 54s	1h 00m 24s			5m 44s	35m 06s	
8	3m 34s	30m 25s			3m 01s	17m 44s	3h 28m 20s
16	1m 47s	15m 26s	3h 15m 45s		1m 37s	9m 13s	1h 47m 51s
32	1m 07s	7m 56s	1h 38m 08s		1m 06s	4m 47s	55m 39s
64		4m 36s	52m 35s			2m 57s	29m 39s
128		3m 02s	30m 41s	4h 47m 22s		2m 53s	25m 23s

We implemented our new parallel codes of Algorithm 5.2 on a Ninf system (Network based Information Library for high performance computing system), which is an infrastructure for world-wide global computing in scientific computation. The basic Ninf system supports client-server based computing, and the computational resources are available as remote libraries at a remote computation host. The remote libraries can be called through the global network from a programmer's client program. For further details on Ninf system, the reader should refer to the articles [9, 10]. We ran the parallel codes on a PC cluster, which consists of 64-nodes connected via a 100 Base-T network, which has a peak unidirectional bandwidth of 100 MB per second. Each node on the PC cluster is a Intel Pentium III 824MHz dual-CPU with 640MB memory. We use eight different Ninf configurations with N processors, by varying the number of N between 1 and 128.

Corresponding to two kinds of benchmark polynomial systems, Table 2 shows computational time on N parallel processors. In these tables, the empty entries show that the parallel algorithm requires more than 5 hours or less than 1 minute to obtain all solutions of Problem 2.1'. To the best of our knowledge, the record of the largest n for cyclic n-roots problems is 13. T.Y. Li and X. Li [7] achieved the record of cyclic13 with 28h 3m 5s computational time on 400MHz Intel Pentium II CPU with 256 MB of memory.

6 Concluding Remarks

In this paper, we have discussed how to allocate mixed cells efficiently. Those cells play a crucial role in constructing homotopies $\boldsymbol{H}(\boldsymbol{x}, t) = \boldsymbol{0}$ within the polyhedral homotopy method. In conclusion, (i) our algorithm saves computation to solve subproblems with a sensitivity technique of linear programs, and (ii) we implemented our algorithm on parallel computing system. For the sake of such achievements, we can handle larger polynomial systems, which no existing methods can deal with.

As further research, we consider the following issues; (a) devise the sensitivity technique which enjoys both advantages of primal and dual approaches described in Section 4, (b) combine our algorithm with some path following procedure for homotopy curves, and (c) provide a whole algorithm of the polyhedral homotopy method for parallel computing. In this study, we have implemented the step which constructs start systems $\boldsymbol{Q}(\boldsymbol{x}) = \boldsymbol{0}$ and homotopies $\boldsymbol{H}(\boldsymbol{x}, t) = \boldsymbol{0}$ on parallel computing system. Now, we are trying to device a method for following each homotopy path in parallel. One aim of future research is to provide a software package for finding all isolated solutions of a polynomial system in a totally parallel manner, and deal with larger dimensional polynomial systems with less computational time.

Acknowledgment: The authors would like to thank Professor T.Y. Li of Michigan State University. He introduced the mixed cell problem of this paper to us, and we learned his studies related to the polyhedral homotopy method in detail when he visited us in July 2000.

References

[1] D.N. Bernstein, "The number of roots of a system of equations," *Functional Analysis and Appl.* **9**(3) (1975) 183–185.

[2] G. Björck and R. Fröberg, "A faster way to count the solutions of inhomogeneous systems of algebraic equations, with applications to cyclic n-roots", *Journal Symbolic Computation* **12** (1991) 329-336.

[3] I.Z. Emiris and J.F. Canny, "Efficient incremental algorithms for the sparse resultant and the mixed volume," *Journal of Symbolic Computation* **20** (1995) 117–149.

[4] T. Gao and T.Y. Li, "Mixed Volume Computation Via Linear Programming," *submitted.*

[5] B. Huber and B. Sturmfels, "A Polyhedral method for solving sparse polynomial systems," *Mathematics of Computation* **64** (1995) 1541–1555.

[6] T.Y. Li, "Solving polynomial systems by polyhedral homotopies", *Taiwan Journal of Mathematics* **3** (1999) 251-279.

[7] T.Y. Li and X. Li, "Finding Mixed Cells in the Mixed Volume Computation," *submitted.*

[8] A. Morgan, *Solving polynomial systems using continuation for engineering and scientific problems,* Prentice-Hall, New Jersey, 1987.

[9] M. Sato, H. Nakada, S. Sekiguchi, S. Matsuoka, U. Nagashima and H. Takagi, "Ninf: a network based information library for a global world-wide computing infrastructure," in *Lecture Note in Computer Science, High-Performance Computation and Network* (1997) 491–502.

[10] S. Sekiguchi, M. Sato, H. Nakada, S. Matsuoka and U. Nagashima , "– Ninf –: network based information library for globally high performance computing," in *Proc. of Parallel Object-Oriented Methods and Applications (POOMA),* February 1996.

[11] J. Verschelde, "Homotopy continuation methods for solving polynomial systems," *Ph.D. thesis, Department of Computer Science, Katholieke Universiteit Leuven,* Belgium, 1996.

[12] J. Verschelde, "PHCPACK: A general-purpose solver for polynomial systems by homotopy continuation," *ACM Transactions on Mathematical Software* **25** (1999) 251-276.