Research Reports on Mathematical and Computing Sciences
Series B : Operations Research

Department of Mathematical and Computing Sciences
Tokyo Institute of Technology
2-12-1 Oh-Okayama, Meguro-ku, Tokyo 152-8552 Japan

Enumeration of All Solutions of a Combinatorial Linear Inequality System Arising from the Polyhedral Homotopy Continuation Method

Akiko Takeda (*takeda@is.titech.ac.jp*)

Masakazu Kojima (*kojima@is.titech.ac.jp*)
Department of Mathematical and Computing Sciences, Tokyo Institute of Technology
2-12-1 Oh-Okayama, Meguro-ku, Tokyo 152-8552 Japan

Katsuki Fujisawa (*fujisawa@is-mj.archi.kyoto-u.ac.jp*)
Dept. of Architecture and Architectural Systems, Kyoto University
Yoshida-Honmachi, Sakyo-ku, Kyoto 606-8501 Japan

November 2000

Abstract An interesting combinatorial (enumeration) problem arises in the initial phase of the polyhedral homotopy continuation method for computing all solutions of a polynomial equation system in complex variables. It is formulated as a problem of finding all solutions of a specially structured system of linear inequalities with a certain additional combinatorial condition. This paper presents a computational method for the problem fully utilizing the duality theory and the simplex method for linear programs, and reports numerical results on a single CPU implementation and a parallel implementation of the method.

Key words. Polynomial System, Mixed Volume, Polyhedral Homotopy, Linear Inequality, Linear Program, Simplex Method, Enumeration, Parallel Computation

1 Introduction

Let $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})) = \boldsymbol{0}$ be a system of n polynomial equations in n complex unknowns $x_i \in \mathrm{C}$ $(i = 1, 2, \ldots, n)$, where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \mathrm{C}^n$. We denote each monomial as $\boldsymbol{x}^a = x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n} \in \mathrm{C}$, and identify it with a lattice point $\boldsymbol{a} = (a_1, a_2, \ldots, a_n) \in \mathrm{Z}_+^n \equiv \{0, 1, 2, \ldots\}^n$. Denoting each term of the ith equation as $c_{i,a}\, \boldsymbol{x}^a \in \mathrm{C}$ with a coefficient $c_{i,a} \in \mathrm{C}$ and $\boldsymbol{a} \in \mathrm{Z}_+^n$, we then write each polynomial $f_i(\boldsymbol{x})$ as

$$f_i(\boldsymbol{x}) = \sum_{\boldsymbol{a} \in \mathcal{A}_i} c_{i,a}\, \boldsymbol{x}^a \quad (i = 1, 2, \ldots, n). \tag{1}$$

Here $\mathcal{A}_i$ $(i = 1, \ldots, n)$ is a set of lattice points corresponding to the terms of the ith equation, and is called the *support* of the polynomial $f_i(\boldsymbol{x})$. The purpose of this article is to present an efficient computational method for a combinatorial linear inequality system (*i.e.*, a linear inequality system with an additional combinatorial condition) which arises as an initial phase of the *polyhedral homotopy continuation method* [10, 11] for computing all isolated solutions of a polynomial equation system $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$.

A homotopy continuation method constructs an auxiliary starting polynomial equation system $\boldsymbol{g}(\boldsymbol{x}) = (g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \ldots, g_n(\boldsymbol{x})) = \boldsymbol{0}$ and a family of polynomial equation systems $\boldsymbol{h}(\boldsymbol{x}, t) = \boldsymbol{0}$ with a parameter $t \in [0, 1]$, which satisfy the following properties:

(i) $\boldsymbol{h} : \mathrm{C}^n \times [0, 1] \to \mathrm{C}^n$ is a continuous mapping such that $\boldsymbol{h}(\cdot, 0) = \boldsymbol{g}(\cdot)$ and $\boldsymbol{h}(\cdot, 1) = \boldsymbol{f}(\cdot)$. In view of this property, we will call $\boldsymbol{h}$ a *homotopy* between $\boldsymbol{g}$ and $\boldsymbol{f}$.

(ii) The solutions of $\boldsymbol{h}(\boldsymbol{x}, 0) \equiv \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$, say $\bar{\boldsymbol{x}}^1, \bar{\boldsymbol{x}}^2, \ldots, \bar{\boldsymbol{x}}^\ell$, are known.

(iii) The connected component of the solution set $\{(\boldsymbol{x}, t) \in \mathrm{C}^n \times [0, 1) : \boldsymbol{h}(\boldsymbol{x}, t) = 0\}$ containing each $(\bar{\boldsymbol{x}}^j, 0)$ forms a smooth curve (*homotopy path*) $\{(\boldsymbol{\phi}^j(t), t) : t \in [0, 1)\}$ such that $\boldsymbol{\phi}^j(0) = \bar{\boldsymbol{x}}^j$ and that if $\boldsymbol{\phi}^j(t)$ converges to some $\hat{\boldsymbol{x}}^j$ as $t \to 1$ then $\hat{\boldsymbol{x}}^j$ is a solution of $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$ $(j = 1, 2, \ldots, \ell)$.

Starting from each known $(\boldsymbol{\phi}^j(0), 0) = (\bar{\boldsymbol{x}}^j, 0)$, we then trace the homotopy path $\{(\boldsymbol{\phi}^j(t), t) : t \in [0, 1)\}$ numerically. If the homotopy path is bounded, we obtain an approximation of a solution of $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$ when we reach $t = 1$. But the path may be unbounded. In this case, the amount of work which has been done to trace the path turns out to be vain efforts. One of the key and important issues in homotopy continuation methods is how we create fewer homotopy paths including a valid set of homotopy paths, a set of bounded homotopy paths that lead to all isolated solutions of $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$. Ideally, we want to create a minimal valid set of homotopy paths; in this case the number of homotopy paths in the set coincides with the number of isolated solutions of $\boldsymbol{f}(\boldsymbol{x})$.

It is known [10, 11] that the polyhedral homotopy continuation method generates much fewer homotopy paths including a valid set of homotopy paths than classical linear homotopy continuation method [1, 4, 7, 8, 9]. In the former method, we construct multiple pairs of an auxiliary starting polynomial equation system $\boldsymbol{g}^p(\boldsymbol{x}) = \boldsymbol{0}$ and a homotopy polynomial equation system $\boldsymbol{h}^p(\boldsymbol{x}, t) = \boldsymbol{0}$ $(p = 1, 2, \ldots)$ such that the homotopy paths induced from them are bounded ones, and that they together cover all solutions of $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$. The construction of such pairs of $\boldsymbol{g}^p(\boldsymbol{x}) = \mathbf{0}$ and $\boldsymbol{h}^p(\boldsymbol{x}, t) = \mathbf{0}$ $(p = 1, 2, \ldots)$, however, is not an easy task at all. Indeed, it requires to compute all *fine mixed cells* of the polynomial equation system $\boldsymbol{f}(\boldsymbol{x}) = \mathbf{0}$, and it takes major computational work in the polyhedral homotopy continuation method. This problem is formulated as computing all solutions of a combinatorial linear inequality system. See [2, 3, 5, 6, 10, 11, 12, 16] for more details.

This article presents an efficient computational method for the combinatorial linear inequality system mentioned above. Recently Li and Li [12] proposed a computational method for the same problem. Both methods are similar to each other in the sense that they use the simplex method for linear programs. Main differences between the two methods are:

- Our method fully utilizes the duality theory of linear programs.
- Our method is explicitly described as a depth-first search over an enumeration tree in the framework of the branch-and-bound method, which makes it easier to understand a fundamental structure of the problem and its solution for the researchers in the field of mathematical programming.
- Our method fits parallel implementation quite nicely.

In Section 2, we describe our problem of computing all solutions of a combinatorial linear inequality system in details. Sections 3 and 4 are devoted to some basic materials that are necessary to describe our method. Section 3 introduces a family of linear inequality subsystems of the combinatorial linear inequality system, and we embed them in an enumeration tree. We will see there that each feasible leaf node is corresponding to a solution of the combinatorial linear inequality system and vice versa, and that if a node is infeasible then so are its children nodes. Section 4 provides some tests to determine whether a node of the enumeration tree is either feasible or infeasible with the use of the duality theorem and the simplex method for linear programs. We present our method as a depth-first search to the enumeration tree and its parallel implementation in Section 5, and numerical results on a single CPU implementation and a parallel implementation of the method applied to two types of benchmark problems in Section 6. From the numerical results, we will see that our method is as efficient as the state-of-art method given by Li and Li [12], and that the parallel implementation of our method is really powerful; it can solve a large size problem (the cyclic-14 problem), which have not been solved before, in less than five hours using a PC cluster system which has 128 Intel Pentium III CPUs.

2 A combinatorial linear inequality system

According to the paper [10], we distinguish three cases of the polynomial system (1). The polynomial system (1) is *unmixed* when all the supports $\mathcal{A}_i$ $(i = 1, \ldots, n)$ are equal to each other; *fully mixed* when they are all distinct; and *semi-mixed* otherwise. In Section 2, 3 and 4, we will focus our attention on a semi-mixed polynomial system; mixed and unmixed polynomial systems are treated as its special cases. We may assume without loss of generality that the first m supports $\mathcal{A}_i$ $(i = 1, 2, \ldots, m)$ differ each other, and that any of the last

$n-m$ supports $\mathcal{A}_i$ ($i = m+1, \dots, n$) coincides with some of the first m supports. Here m is a positive integer not greater than n. For simplicity of notation, we use the symbol M for the index set $\{1, 2, \dots, m\}$. For each $j \in M$, let s_j denote the number of the supports among $\mathcal{A}_i$ ($i = 1, \dots, n$) which coincide with $\mathcal{A}_j$, including $\mathcal{A}_j$ itself. Then s_j ($j = 1, \dots, m$) are positive integers such that $\sum_{j=1}^m s_j = n$. Let $\langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle$ denote the inner product of two vectors $\boldsymbol{a}$ and $\boldsymbol{\alpha}$ in the n-dimensional Euclidean space R^n, and $\sharp C$ the number of elements in C. For every $\boldsymbol{a} \in \mathcal{A}_j$ and $j \in M$, let $\omega_j(\boldsymbol{a})$ be a real number chosen generically.

Problem 2.1. Find a solution $(\boldsymbol{\alpha}, \boldsymbol{\beta}) = (\alpha_1, \alpha_2, \dots, \alpha_n, \beta_1, \beta_2, \dots, \beta_m) \in \mathrm{R}^{n+m}$ which satisfies

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

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

Throughout the paper we assume:

Condition 2.2. At most $n + m$ equalities hold for any solution of the linear inequality system (2).

This condition is a standard nondegeneracy condition which is often imposed on linear programs to make a description of the simplex method simpler and easier. Note that the condition holds generically if we choose the constant scalars $\omega_j(\boldsymbol{a})$ ($\boldsymbol{a} \in \mathcal{A}_j,\ j \in M$) of (2) randomly. Under the condition, we may replace "exactly" in the statement of Problem 2.1 by "at least".

Define a polyhedral set

$$V = \{(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathrm{R}^{n+m} : \beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \le \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in \mathcal{A}_j,\ j \in M)\}.$$

Then each solution of the linear inequality system (2) with exactly $n + m$ equalities corresponds to a vertex of V and vice versa. Therefore we can restate Problem 2.1 as: Find a vertex $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of V such that exactly (or at least) $(s_j + 1)$ equalities hold for each $j \in M$.

Suppose that $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ be a solution of Problem 2.1. Let

$$\begin{aligned} C_j &= \{\boldsymbol{a} \in \mathcal{A}_j : \beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a})\}, \\ \boldsymbol{C} &= (C_1, C_2, \dots, C_m) \in \mathcal{A} \equiv \prod_{j=1}^m \mathcal{A}_j. \end{aligned}$$

Then

$$\begin{aligned} &\sharp C_j = s_j + 1, \\ &\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in C_j,\ j \in M), \\ &\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle < \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in \mathcal{A}_j \backslash C_j,\ j \in M). \end{aligned}$$

We call $\boldsymbol{C}$ a *fine mixed cell* of $\mathcal{A}$ induced by a lifting $\boldsymbol{\omega} = (\omega_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in M)$. Here $(\omega_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in M)$ denotes a vector consisting of $\omega_j(\boldsymbol{a})$ ($\boldsymbol{a} \in \mathcal{A}_j,\ j \in M$).

A collection of all fine mixed cells, which are induced from all solutions of Problem 2.1, is called a *fine mixed subdivision* of $\mathcal{A}$. In the polyhedral homotopy continuation method, it is a key issue how efficiently we compute a fine mixed subdivision of $\mathcal{A}$ of Problem 2.1. Assuming that the number of all solutions for Problem 2.1 is q, let $\boldsymbol{C}^p = (C_1^p, C_2^p, \ldots, C_m^p)$ $(p = 1, \ldots, q)$ be all mixed cells, and $\mathcal{S} = \{\boldsymbol{C}^p : p = 1, 2, \ldots, q\}$ a fine mixed subdivision of $\mathcal{A}$. We use each fine mixed cell $\boldsymbol{C}^p$ to construct a pair of an auxiliary starting polynomial equation system $\boldsymbol{g}^p(\boldsymbol{x}) = \boldsymbol{0}$ and a homotopy polynomial equation system $\boldsymbol{h}^p(\boldsymbol{x}, t) = \boldsymbol{0}$ $(p = 1, 2, \ldots, q)$ mentioned in the Introduction. For more details on how we use a fine mixed subdivision $\mathcal{S}$ in the polyhedral homotopy method, see the articles [5, 6, 10, 11, 12, 16, etc].

In the succeeding two sections, we present a numerical method for computing all solutions of Problem 2.1.

3 A family of linear inequality subsystems and an enumeration tree

One easy and primitive method for computing all solutions of Problem 2.1 is as follows. First, prepare a set of all possible candidates for fine mixed cells:

$$\widetilde{\mathcal{S}} = \left\{ \boldsymbol{C} = (C_1, C_2, \ldots, C_m) : \quad C_j \subseteq \mathcal{A}_j, \quad \sharp C_j = s_j + 1 \ (j \in M) \ \right\}.$$

Then, for each $\boldsymbol{C} \in \widetilde{\mathcal{S}}$, solve the linear equation system

$$\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in C_j, \ j \in M), \tag{3}$$

and check whether the solution $(\bar{\boldsymbol{\alpha}}, \bar{\boldsymbol{\beta}}) \in \mathrm{R}^{n+m}$ of (3) is feasible for the remaining linear inequalities

$$\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle < \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in \mathcal{A}_j \backslash C_j, \ j \in M).$$

As a given polynomial system (1) involves a larger number of complex variables and/or a larger number of monomials, however, the number of elements of $\widetilde{\mathcal{S}}$ increases rapidly. In such a case, we are required to perform a huge number of feasibility tests in this primitive method, so that the method becomes quite inefficient and impractical. We will present a practical enumeration technique to avoid such exhaustive feasibility tests for all possible candidate linear systems.

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

$$\ell = m, \ K^0 = \emptyset, \ K^p = \{1, 2, \ldots, p\} \ (p = 1, 2, \ldots, m). \tag{4}$$

3.1 A family of linear inequality subsystems

Let K be a nonempty subset of M. Later we will specify $K = K^p$ $(p = 1, 2, \ldots, \ell)$. For each $\boldsymbol{F} = (F_j : j \in K)$ with $F_j \subseteq \mathcal{A}_j$, we consider a linear inequality subsystem

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

Here $(F_j : j \in K)$ denotes a vector of sets F_j $(j \in K)$. Define

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

and assume that $\mathcal{F}^{\leq}(\emptyset) = \mathcal{F}^{=}(\emptyset) = \mathcal{F}^{=\&\mathrm{f}}(\emptyset) = \{\emptyset\}$ for convention. By definition, we see that

$$\mathcal{F}^{=\&\mathrm{f}}(K) \subseteq \mathcal{F}^{=}(K) \subseteq \mathcal{F}^{\leq}(K) \ \text{ for every } K \subseteq M.$$

Now, computing all solutions of Problem 2.1 is reduced to computing a solution of $\mathrm{E}(M, \boldsymbol{F})$ for every $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$. Condition 2.2 ensures that the solution set of each $\mathrm{E}(M, \boldsymbol{F})$ with an $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$ consists of a single vertex of V. We also note that the set $\mathcal{F}^{=\&\mathrm{f}}(M)$ coincides with a mixed subdivision $\mathcal{S}$ of $\mathcal{A}$ introduced in the previous section. Thus we want to efficiently enumerate all $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$. In particular, we need to avoid brutal exhausting feasibility tests of $\mathrm{E}(M, \boldsymbol{F})$ for every $\boldsymbol{F} \in \mathcal{F}^{=}(M)$. The lemma below plays an essential role to reduce the number of feasibility tests.

Lemma 3.1. *Let $1 \leq p \leq q \leq \ell$, $\boldsymbol{F}^p \in \mathcal{F}^{\leq}(K^p)$, $\boldsymbol{F}^q \in \mathcal{F}^{\leq}(K^q)$, and $F^p_j \subseteq F^q_j$ $(j \in K^p)$. If $E(K^p, \boldsymbol{F}^p)$ is infeasible, then so is $E(K^q, \boldsymbol{F}^q)$.*

Proof: All constraints of $\mathrm{E}(K^p, \boldsymbol{F}^p)$ are included in $\mathrm{E}(K^q, \boldsymbol{F}^q)$. Therefore, if $\mathrm{E}(K^p, \boldsymbol{F}^p)$ is infeasible, then so is $\mathrm{E}(K^q, \boldsymbol{F}^q)$. ∎

This lemma implies that if we find that $\mathrm{E}(K^p, \boldsymbol{F}^p)$ with $\boldsymbol{F}^p \in \mathcal{F}^{=}(K^p)$ is infeasible, we can omit feasibility tests for all $\mathrm{E}(M, \boldsymbol{F})$ with $\boldsymbol{F} \in \mathcal{F}^{=}(M)$ satisfying $F_j = F^p_j$ $(j \in K^p)$. Hence we can considerably save computation required for feasibility tests when $\sharp K^p$ is less than $\sharp M (= m)$. In the next subsection, we will embed a tree structure in the family of subsets $\boldsymbol{F} \in \mathcal{F}^{=}(K^p)$ $(p = 0, 1, 2, \ldots, \ell)$.

Let $\boldsymbol{F} \in \mathcal{F}^{\leq}(K)$. None of the linear inequalities

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

involved in the original linear inequality system (2) are included in $\mathrm{E}(K, \boldsymbol{F})$. Those inequalities never affect the feasibility nor the infeasibility of $\mathrm{E}(K, \boldsymbol{F})$. That is, the set of linear inequalities in (6) is irrelevant to checking the feasibility and/or the infeasibility of $\mathrm{E}(K, \boldsymbol{F})$. In fact, 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 every $j \in M \backslash K$ so that the extended solution $(\widetilde{\boldsymbol{\alpha}}, \widetilde{\beta}_j : j \in M)$ satisfies all the inequalities in (6).

3.2 Embedding a tree structure

In order to enumerate all $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$, we build an enumeration tree in 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 $\boldsymbol{F}' \in \mathcal{F}^{=}(K^{p+1})$ in the $(p+1)$th level is a *child* of a node $\boldsymbol{F} \in \mathcal{F}^{=}(K^p)$ in the pth level if and only if $\boldsymbol{F}' = (\boldsymbol{F}, F_j : j \in K^{p+1} \backslash K^p)$. 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. We call a node $\boldsymbol{F}' \in \mathcal{F}^{=}(K^q)$ in the qth level with $q > p$ a *descendant* of a node $\boldsymbol{F} \in \mathcal{F}^{=}(K^p)$ in the pth level if $\boldsymbol{F}' = (\boldsymbol{F}, F_j : j \in K^q \backslash K^p)$. In this case, there is a unique sequence $\{\boldsymbol{F}^r \in \mathcal{F}^{=}(K^r) : r = p, p+1, \ldots, q\}$ such that $\boldsymbol{F}^p = \boldsymbol{F}$, $\boldsymbol{F}^q = \boldsymbol{F}'$ and that each $\boldsymbol{F}^{r+1} \in \mathcal{F}^{=}(K^{r+1})$ is a child node of $\boldsymbol{F}^r \in \mathcal{F}^{=}(K^r)$. In fact, the sequence is given by

$$\boldsymbol{F}^p = \boldsymbol{F}, \ \boldsymbol{F}^{r+1} = (\boldsymbol{F}^r, F_j' : j \in K^{r+1} \backslash K^r) \in \mathcal{F}^{=}(K^{r+1}) \quad (r = p, p+1, \ldots, q-1).$$

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 apply the depth-first search to the enumeration tree that we have built above. We know that $\mathrm{E}(K^0, \emptyset)$ associated with the root node is feasible, so that 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.*, $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K^p)$, 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 infeasible from Lemma 3.1; hence its descendants never contain any feasible node. In this case, we can terminate the node $\boldsymbol{F} \in \mathcal{F}^{=}(K^p)$ in the pth level, and we 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 nodes. If $\mathrm{E}(K^p, \boldsymbol{F})$ is feasible and $p = \ell$, then we obtain one desired node in $\mathcal{F}^{=\&\mathrm{f}}(M)$. Continuing this enumeration procedure, we eventually generate all nodes in $\cup_{p=1}^{\ell} \mathcal{F}^{=\&\mathrm{f}}(K^p)$.

We can easily adapt this framework to parallel computation. Taking some $p \geq 1$, we consider the family of linear inequality systems $\mathrm{E}(K^p, \boldsymbol{F})$ $(\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K^p))$. Then the number of systems in the family amounts to $\sharp \mathcal{F}^{=\&\mathrm{f}}(K^p)$. Each linear inequality system $\mathrm{E}(K^p, \boldsymbol{F})$ in the family naturally induces a subtree with a root node $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K^p)$. We allocate all such linear inequality systems among multiple processors, and each processor performs the depth first search described above to an assigned subtree with a root node $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(K^p)$ to compute solutions $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathrm{R}^{n+m}$ of $\mathrm{E}(M, \boldsymbol{F}')$ for all descendant leaf nodes $\boldsymbol{F}' \in \mathcal{F}^{=\&\mathrm{f}}(M)$ of $\boldsymbol{F}$. Summing up all solutions from each processor, we obtain all solutions of Problem 2.1. For effective parallel computation, we need to balance the number of processors, the number of the linear inequality systems to be allocated and the size of each linear inequality system by choosing the depth p of the original enumeration tree appropriately.

4 Reformulation via linear programs

Corresponding to each linear inequality 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 primal-dual pair of linear programs:

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

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

The constraint linear inequalities of the dual problem $\mathrm{D}(K, \boldsymbol{F})$ are exactly the same as those of $\mathrm{E}(K, \boldsymbol{F})$. Hence, the linear inequality system $\mathrm{E}(K, \boldsymbol{F})$ is feasible if and only if the dual problem $\mathrm{D}(K, \boldsymbol{F})$ is feasible, and each solution of Problem 2.1 is corresponding to a solution of $\mathrm{D}(M, \boldsymbol{F})$ with some $\boldsymbol{F} \in \mathcal{F}^{=\&\mathrm{f}}(M)$ and vice versa. It should be also noted that the constraint linear inequalities

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

and the constraint linear equalities

$$\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha}\rangle = \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in F_j,\ j \in K)$$

in the dual problem $\mathrm{D}(K, \boldsymbol{F})$ are corresponding to the nonnegative variables

$$x_j(\boldsymbol{a}) \geq 0 \quad (\boldsymbol{a} \in \mathcal{A}_j \backslash F_j,\ j \in K)$$

and the free variables

$$-\infty < x_j(\boldsymbol{a}) < +\infty \quad (\boldsymbol{a} \in F_j,\ j \in K)$$

in the primal problem $\mathrm{P}(K, \boldsymbol{F})$, respectively.

In the remainder of this paper, we impose the standard nondegeneracy conditions on the primal-dual pair of linear programs $\mathrm{P}(K, \boldsymbol{F})$ and $\mathrm{D}(K, \boldsymbol{F})$.

Condition 4.1.

(a) If $\boldsymbol{x} = (x_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K)$ is a feasible solution of the primal problem $\mathrm{P}(K, \boldsymbol{F})$ then at least $n + \sharp K$ elements of $\boldsymbol{x}$ are nonzero.

(b) If $(\boldsymbol{\alpha}, \beta_j : j \in K)$ is a feasible solution of the dual problem $\mathrm{D}(K, \boldsymbol{F})$ then at most $n + \sharp K$ equalities hold in its constraints.

Recall that the constant real numbers $\omega_j(\boldsymbol{a})$ $(\boldsymbol{a} \in \mathcal{A}_j,\ j = 1, 2, \ldots, \ell)$ were chosen generically, so that the condition (b) is justified. In order to apply the duality theory between $\mathrm{P}(K, \boldsymbol{F})$ and $\mathrm{D}(K, \boldsymbol{F})$ effectively, we need to make the primal problem feasible by choosing $\boldsymbol{d}$ and b_j $(j \in K)$ appropriately, although we can take any linear function as the objective function in $\mathrm{D}(K, \boldsymbol{F})$. We show how to choose and update such $\boldsymbol{d} \in \mathrm{R}^n$ and $b_j \in \mathrm{R}$ $(j = 1, 2, \ldots, \ell)$ in a generic way in Section 4.4, so that the condition (a) above will be also justified too.

If we apply the (standard) primal simplex method to the dual problem $\mathrm{D}(K, \boldsymbol{F})$, we need to convert each of the inequality constraints in $\mathrm{D}(K, \boldsymbol{F})$ to the equality form by introducing a nonnegative slack variable. Such a transformation increases the number of variables considerably, and moreover, the size of a basis matrix of the transformed dual problem turns out to be $\sum_{j \in K} \sharp \mathcal{A}_j$ which is much larger than the size $n + \sharp K$ $(\leq 2n)$ of a basis matrix of the primal problem $\mathrm{P}(K, \boldsymbol{F})$. Therefore it would not be efficient to apply the primal simplex method to the transformed dual problem. Instead we had better to apply the primal or dual simplex method to the primal problem $\mathrm{P}(K, \boldsymbol{F})$ which requires no additional slack variables because it is already an equality standard form with some free variables.

Let $p \in \{0, 1, 2, \ldots, \ell - 1\}$. Suppose that $\boldsymbol{F}^p \in \mathcal{F}^{=\&\mathrm{f}}(K^p)$. In the remainder of this section, we will discuss various tests to check whether some of the child nodes of $\boldsymbol{F}^p$, *i.e.*, some of the linear inequality systems $\mathrm{E}(K^{p+1}, \boldsymbol{F})$ with $\boldsymbol{F} = (\boldsymbol{F}^p, F_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{=}(K^{p+1})$ are feasible or infeasible.

4.1 Infeasibility test based on dual problems

Let $F_j = \emptyset$ $(j \in K^{p+1} \backslash K^p)$ and $\boldsymbol{F} = (\boldsymbol{F}^p, F_j : j \in K^{p+1} \backslash K^p)$. Choose a $\boldsymbol{G} \in \mathcal{F}^{\leq}(K^{p+1})$ such that

$$\boldsymbol{G} = (\boldsymbol{F}^p, G_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{\leq}(K^{p+1}) \text{ and } G_j \neq \emptyset \text{ for some } j \in K^{p+1} \backslash K^p.$$

Using randomly generated numbers $x_j(\bar{\boldsymbol{a}}) > 0$ $(\bar{\boldsymbol{a}} \in G_j,\ j \in K^{p+1})$, define the coefficients of the objective function in $\mathrm{D}(K^{p+1}, \boldsymbol{F})$ as

$$b_j = \sum_{\bar{\boldsymbol{a}} \in G_j} x_j(\bar{\boldsymbol{a}}) \ \ (j \in K^{p+1}) \ \text{ and } \boldsymbol{d} = \sum_{j \in K^{p+1}} \sum_{\bar{\boldsymbol{a}} \in G_j} \bar{\boldsymbol{a}} \ x_j(\bar{\boldsymbol{a}}). \tag{7}$$

The objective function can be rewritten as

$$\sum_{j \in K^{p+1}} b_j \beta_j - \langle \boldsymbol{d}, \boldsymbol{\alpha} \rangle = \sum_{j \in K^{p+1}} \sum_{\bar{\boldsymbol{a}} \in G_j} (\beta_j - \langle \bar{\boldsymbol{a}}, \boldsymbol{\alpha} \rangle) \ x_j(\bar{\boldsymbol{a}}), \tag{8}$$

so we see that the objective value is bounded from above because

$$\sum_{j \in K^{p+1}} b_j \beta_j - \langle \boldsymbol{d}, \boldsymbol{\alpha} \rangle \leq \sum_{j \in K^{p+1}} \sum_{\bar{\boldsymbol{a}} \in G_j} \omega_j(\bar{\boldsymbol{a}}) \ x_j(\bar{\boldsymbol{a}}).$$

Also $\mathrm{D}(K^{p+1}, \boldsymbol{F})$ is feasible. In fact, if we define

$$\hat{\beta}_j = \min\{\langle \boldsymbol{a}, \hat{\boldsymbol{\alpha}} \rangle + \omega(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j\} \quad (j \in K^{p+1} \backslash K^p),$$

where $(\hat{\boldsymbol{\alpha}}, \hat{\beta}_j : j \in K^p)$ denotes a feasible solution of $\mathrm{D}(K^p, \boldsymbol{F}^p)$, then $(\hat{\boldsymbol{\alpha}}, \hat{\beta}_j : j \in K^{p+1})$ turns out to be a feasible solution $\mathrm{D}(K^{p+1}, \boldsymbol{F})$. Therefore the dual problem $\mathrm{D}(K^{p+1}, \boldsymbol{F})$ has an optimal solution $(\boldsymbol{\alpha}^*, \beta_j^* : j \in K^{p+1})$.

Lemma 4.2. *(Infeasibility test based on dual problems) Assume that*

$$\sum_{j \in K^{p+1}} b_j \beta_j^* - \langle \boldsymbol{d}, \boldsymbol{\alpha}^* \rangle < \sum_{j \in K^{p+1}} \sum_{\bar{\boldsymbol{a}} \in G_j} \omega_j(\bar{\boldsymbol{a}})\ x_j(\bar{\boldsymbol{a}}). \tag{9}$$

Then any linear inequality system $E(K^{p+1}, \widetilde{\boldsymbol{F}})$ with $\widetilde{\boldsymbol{F}} = (\boldsymbol{F}^p, \widetilde{F}_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{=}(K^{p+1})$ and $\widetilde{F}_j \supseteq G_j$ $(j \in K^{p+1} \backslash K^p)$ is infeasible.

Proof: Since the problem $\mathrm{D}(K^{p+1}, \boldsymbol{F})$ maximizes the objective function of (8), the strict inequality in (9) means $\beta_j - \langle \bar{\boldsymbol{a}}, \boldsymbol{\alpha} \rangle < \omega_j(\bar{\boldsymbol{a}})$ for some $\bar{\boldsymbol{a}} \in G_j$ and some $j \in K^{p+1} \backslash K^p$, *i.e.*, the infeasibility of $\mathrm{D}(K^{p+1}, \boldsymbol{G})$. It also means the infeasibility of $\mathrm{D}(K^{p+1}, \widetilde{\boldsymbol{F}})$. ∎

It should be noted that some of the linear inequality systems $\mathrm{E}(K^{p+1}, \widetilde{\boldsymbol{F}})$ with $\widetilde{\boldsymbol{F}} = (\boldsymbol{F}^p, \widetilde{F}_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{=}(K^{p+1})$ and $\widetilde{F}_j \supseteq G_j$ $(j \in K^{p+1} \backslash K^p)$ can be infeasible even if the test fails, *i.e.* the strict inequality (9) does not hold. We could further impose on $\boldsymbol{G}$ the condition that $\boldsymbol{G} \in \mathcal{F}^{=}(K^{p+1})$. In this case, the test would completely determine whether a single linear inequality systems $\mathrm{E}(K^{p+1}, \boldsymbol{G})$ is either feasible or infeasible.

4.2 Feasibility test based on dual problems

Choose a $\boldsymbol{G} \in \mathcal{F}^{\leq}(K^{p+1})$ such that $G_j = F_j^p$ $(j \in K^p)$. We consider the dual problem $\mathrm{D}(K^{p+1}, \boldsymbol{G})$ with an arbitrary $b_j \in \mathrm{R}$ $(j \in K^{p+1})$ and an arbitrary $\boldsymbol{d} \in \mathrm{R}^n$. To perform the infeasibility test of Lemma 4.2 simultaneously, it is reasonable to choose $b_j \in \mathrm{R}$ $(j \in K^{p+1})$ and $\boldsymbol{d} \in \mathrm{R}^n$ as given in the previous subsection.

Suppose that $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K^{p+1})$ is a basic feasible solution of the dual problem $\mathrm{D}(K^{p+1}, \boldsymbol{G})$. Let

$$\begin{array}{rcl} N_j & = & \{\boldsymbol{a} \in \mathcal{A}_j : \bar{\beta}_j - \langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle < \omega_j(\boldsymbol{a})\} \ \ (j \in K^{p+1}), \\ B_j & = & \{\boldsymbol{a} \in \mathcal{A}_j : \bar{\beta}_j - \langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle = \omega_j(\boldsymbol{a})\} \ \ (j \in K^{p+1}). \end{array} \tag{10}$$

Lemma 4.3. *(Feasibility test based on dual problems) If $\boldsymbol{F} = (F_j : j \in K^{p+1}) \in \mathcal{F}^{\leq}(K^{p+1})$ satisfies $F_j \subseteq B_j$ $(j \in K^{p+1})$, then the linear inequality system $E(K^{p+1}, \boldsymbol{F})$ is feasible.*

Proof: By construction, we see that

$$\begin{array}{l} \bar{\beta}_j - \langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle = \omega_j(\boldsymbol{a}) \ \ (\boldsymbol{a} \in F_j,\ j \in K^{p+1}), \\ \bar{\beta}_j - \langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle \leq \omega_j(\boldsymbol{a}) \ \ (\boldsymbol{a} \in \mathcal{A}_j \backslash F_j,\ j \in K^{p+1}). \end{array}$$

Therefore $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K^{p+1})$ is a feasible solution of $\mathrm{E}(K^{p+1}, \boldsymbol{F})$. ∎

Starting from a basic feasible solution of $\mathrm{D}(K^{p+1}, \boldsymbol{G})$, apply the simplex method to $\mathrm{D}(K^{p+1}, \boldsymbol{G})$ with keeping the feasibility of $\mathrm{D}(K^{p+1}, \boldsymbol{G})$. Then we obtain a sequence of basic feasible solutions of the problem $\mathrm{D}(K^{p+1}, \boldsymbol{G})$. At each basic feasible solution, we can perform the feasibility test of Lemma 4.3 to detect the feasibility of some of the child nodes of $\boldsymbol{F}^p \in \mathcal{F}^{=\&\mathrm{f}}(K^p)$.

Remark 4.4. The one-point and two-point tests given by Li-Li [12] are quite similar to our infeasible and feasible tests based on dual programs which we have described in the previous and this subsections. A minor difference lies in their formulation of dual programs. They dealt with a fully mixed case where $m = n$, $s_j = 1$ $(j = 1, 2, \ldots, n)$, $\ell = n$ and K^q $(q = 0, 1, 2, \ldots, n)$ are chosen as in (4). They took a single element $\boldsymbol{a}$ from $\mathcal{A}_{p+1}$ for G_{p+1} in their one-point test to detect the infeasibility of the linear inequality system $\mathrm{E}(K^{p+1}, (\boldsymbol{F}^p, \{\boldsymbol{a}\}))$, and a pair of two distinct elements $\{\boldsymbol{a}_1, \boldsymbol{a}_2\}$ from $\mathcal{A}_{p+1}$ for G_{p+1} in their two-point test to detect the infeasibility of the linear inequality system $\mathrm{E}(K^{p+1}, (\boldsymbol{F}^p, \{\boldsymbol{a}_1, \boldsymbol{a}_2\}))$. Also they applied the feasibility test at each basic feasible solution of their dual programs.

4.3 Infeasibility and feasibility tests based on primal problems

Let $p \in \{0, 1, 2, \ldots, \ell - 1\}$ and $\boldsymbol{F}^p \in \mathcal{F}^{=\&\mathrm{f}}(K^p)$. We present a test to check whether some of the child nodes of $\boldsymbol{F}^p$ are feasible or infeasible based on primal problems. Choose a $\boldsymbol{G} \in \mathcal{F}^{\leq}(K^{p+1})$ such that

$$\boldsymbol{G} = (\boldsymbol{F}^p, G_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{\leq}(K^{p+1}) \text{ and } G_j \neq \emptyset \text{ for some } j \in K^{p+1} \backslash K^p.$$

We assume for the time being that $b_j \in \mathrm{R}$ $(j \in K^{p+1})$ and $\boldsymbol{d} \in \mathrm{R}^n$ are defined as (7) so that a basic feasible solution $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$ of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$ is available. We will discuss how to construct such $b_j \in \mathrm{R}$ $(j \in K^{p+1})$, $\boldsymbol{d} \in \mathrm{R}^n$ and a basic feasible solution later in the next subsection. For every $j \in K^{p+1}$, let

$$\left.\begin{array}{rcl} B_j & = & \{\bar{\boldsymbol{a}} \in \mathcal{A}_j : \bar{x}_j(\bar{\boldsymbol{a}}) \text{ is a basic variable at } \bar{\boldsymbol{x}}\}, \\ N_j & = & \{\boldsymbol{a} \in \mathcal{A}_j : \bar{x}_j(\boldsymbol{a}) \text{ is a nonbasic variable at } \bar{\boldsymbol{x}}\}. \end{array}\right\} \tag{11}$$

Then we can rewrite the problem $\mathrm{P}(K^{p+1}, \boldsymbol{G})$ as

$$\left.\begin{array}{ll} \text{minimize} & \displaystyle\sum_{j \in K^{p+1}} \sum_{\boldsymbol{a} \in N_j} \bar{\omega}_j(\boldsymbol{a})\ x_j(\boldsymbol{a}) + \bar{\zeta}_0 \\ \text{subject to} & x_i(\bar{\boldsymbol{a}}) = \bar{x}_i(\bar{\boldsymbol{a}}) + \displaystyle\sum_{j \in K^{p+1}} \sum_{\boldsymbol{a} \in N_j} \bar{g}_{ij}(\bar{\boldsymbol{a}}, \boldsymbol{a})\ x_j(\boldsymbol{a}) \quad (\bar{\boldsymbol{a}} \in B_i,\ i \in K^{p+1}), \\ & \quad -\infty < x_j(\boldsymbol{a}) < +\infty \quad (\boldsymbol{a} \in G_j,\ j \in K^{p+1}), \\ & \quad x_j(\boldsymbol{a}) \geq 0 \quad (\boldsymbol{a} \in \mathcal{A}_j \backslash G_j,\ j \in K^{p+1}), \end{array}\right\} \tag{12}$$

where the coefficients

$$\left.\begin{array}{l} \bar{\omega}_j(\boldsymbol{a}) \in \mathrm{R} \quad (\boldsymbol{a} \in N_j,\ j \in K^{p+1}), \\ \bar{\zeta}_0 = \displaystyle\sum_{i \in K^{p+1}} \sum_{\bar{\boldsymbol{a}} \in B_i} \omega_i(\bar{\boldsymbol{a}})\ \bar{x}_i(\bar{\boldsymbol{a}}) \ \in \mathrm{R}, \\ \bar{g}_{ij}(\bar{\boldsymbol{a}}, \boldsymbol{a}) \in \mathrm{R} \quad (\bar{\boldsymbol{a}} \in B_i,\ i \in K^{p+1},\ \boldsymbol{a} \in N_j,\ j \in K^{p+1}), \end{array}\right\} \tag{13}$$

can be obtained from the simplex tableau or the dictionary with the basic feasible solution $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$.

Lemma 4.5. *(Infeasibility test based on primal problems)*
Let $\hat{\boldsymbol{a}} \in N_j$ with some $j \in K^{p+1}$.

(a) Assume that $\bar{\omega}_j(\hat{\boldsymbol{a}}) < 0$. Define $\boldsymbol{F}' = (F'_j : j \in K^{p+1})$ by

$$F'_i \ = \ \{\bar{\boldsymbol{a}} \in B_i : \bar{g}_{ij}(\bar{\boldsymbol{a}}, \hat{\boldsymbol{a}}) < 0\} \quad (i \in K^{p+1}).$$

If $\boldsymbol{F} = (\boldsymbol{F}^p, F_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{=}(K^{p+1})$ and $F'_i \subseteq F_i$ $(i \in K^{p+1})$, then $E(K^{p+1}, \boldsymbol{F})$ is infeasible.

(b) Assume that $\bar{\omega}_j(\hat{\boldsymbol{a}}) > 0$. Define $\boldsymbol{F}' = (F'_j : j \in K^{p+1})$ by

$$\begin{aligned} F'_i &= \{\bar{\boldsymbol{a}} \in B_i : \bar{g}_{ij}(\bar{\boldsymbol{a}}, \hat{\boldsymbol{a}}) > 0\} \quad (i \in K^{p+1},\ i \neq j), \\ F'_j &= \{\bar{\boldsymbol{a}} \in B_j : \bar{g}_{jj}(\bar{\boldsymbol{a}}, \hat{\boldsymbol{a}}) > 0\} \cup \{\hat{\boldsymbol{a}}\}. \end{aligned}$$

If $\boldsymbol{F} = (\boldsymbol{F}^p, F_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{=}(K^{p+1})$ and $F'_i \subseteq F_i$ $(i \in K^{p+1})$, then $E(K^{p+1}, \boldsymbol{F})$ is infeasible.

Proof: (a) If we pivot the nonbasic variable $x_j(\hat{\boldsymbol{a}})$ into a basis by increasing its value from $\bar{x}_j(\hat{\boldsymbol{a}}) = 0$ and keeping the other nonbasic variables zero, the variable $x_j(\hat{\boldsymbol{a}})$ can increase to $+\infty$ and the objective value can decrease to $-\infty$. Since we can always choose $b_j \in \mathrm{R}$ $(j \in K^{p+1})$ and $\boldsymbol{d} \in \mathrm{R}^n$ which appear in the right hand side of the equality and inequality constraints of $\mathrm{P}(K^{p+1}, \boldsymbol{F}')$ so that the problem $\mathrm{P}(K^{p+1}, \boldsymbol{F}')$ is feasible, when the problem $\mathrm{P}(K^{p+1}, \boldsymbol{F}')$ is unbounded we find that the corresponding dual problem $\mathrm{D}(K^{p+1}, \boldsymbol{F}')$ is infeasible. Since $F'_i \subseteq F_i$ for every $i \in K^{p+1}$, we conclude that $\mathrm{D}(K^{p+1}, \boldsymbol{F})$ is infeasible.

(b) By decreasing the nonbasic variable $x_j(\hat{\boldsymbol{a}})$ from its value $\bar{x}_j(\hat{\boldsymbol{a}}) = 0$, we can similarly prove that the problem $\mathrm{P}(K^{p+1}, \boldsymbol{F}')$ is unbounded, and the desired result follows. ∎

Starting from $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$ of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$, apply the simplex method to the problem $\mathrm{P}(K^{p+1}, \boldsymbol{G})$ to generate a sequence of basic feasible solutions of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$. At each basic solution of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$, we can perform the infeasibility test given in Lemma 4.5. After a finite number of pivots, either

(i) we detect that the problem $\mathrm{P}(K^{p+1}, \boldsymbol{G})$ is unbounded,

or

(ii) we have an optimal basic solution $\boldsymbol{x}^* = (x^*_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$ of the problem $\mathrm{P}(K^{p+1}, \boldsymbol{G})$.

Assume that the case (i) occurs. Then we can apply Lemma 4.5 to check the infeasibility of some of the linear inequality systems $\mathrm{E}(K^{p+1}, \boldsymbol{F})$ with $\boldsymbol{F} = (\boldsymbol{F}^p, F_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{=}(K^{p+1})$. In particular, we know that the linear inequality system $\mathrm{E}(K^{p+1}, \boldsymbol{F})$ with $\boldsymbol{F} = (\boldsymbol{F}^p, F_j : j \in K^{p+1} \backslash K^p) \in \mathcal{F}^{=}(K^{p+1})$ and $G_j \subseteq F_j$ $(j \in K^{p+1} \backslash K^p)$ is infeasible.

Now assume that the case (ii) occurs. In this case, we obtain an optimal basic solution $(\boldsymbol{\alpha}^*, \beta^*_j : j \in K^{p+1})$ of the corresponding dual problem $\mathrm{D}(K^{p+1}, \boldsymbol{G})$. Hence we can perform the feasibility test given in Lemma 4.3. In particular, if in addition $\boldsymbol{G} \in \mathcal{F}^{=}(K^{p+1})$, we know the child node $\boldsymbol{G}$ of $\boldsymbol{F}^p$ is feasible.

4.4 Updating basic feasible solutions

Throughout this subsection, we assume that $K^0, K^1, \ldots, K^\ell$ are taken as in (4), and discuss how to update the basis information when we proceed from the pth level of the enumeration tree to the $(p+1)$st level. Let $p \in \{1, 2, \ldots, \ell - 1\}$. Suppose that $\boldsymbol{F}^p \in \mathcal{F}^{=\&\mathrm{f}}(K^p)$. Then the linear inequality system $\mathrm{E}(K^p, \boldsymbol{F}^p)$ is feasible. We may assume that the feasibility of $\mathrm{E}(K^p, \boldsymbol{F}^p)$ has been detected either when we find a basic feasible solution of $\mathrm{D}(K^p, \boldsymbol{F}^p)$ by the feasibility test on dual problems given in Lemma 4.3 or when we obtain an optimal basic solution of $\mathrm{P}(K^p, \boldsymbol{F}^p)$ as mentioned in the last paragraph of Section 4.3. In both cases, we obtain a basic feasible solution $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K^p)$ of $\mathrm{D}(K^p, \boldsymbol{F}^p)$, which satisfies the relations

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

(Recall Condition 4.1). We will construct an optimal basic solution of $\mathrm{D}(K^{p+1}, \boldsymbol{G})$ and an optimal basic solution of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$ for some $\boldsymbol{d} \in \mathrm{R}^n$, some $b_j \in \mathrm{R}$ $(j \in K^{p+1})$, and some $\boldsymbol{G} = (\boldsymbol{F}^p, G_{p+1}) \in \mathcal{F}^{\leq}(K^{p+1})$. Let

$$\begin{array}{rcl}
\bar{\beta}_{p+1} & = & \min\{\langle \boldsymbol{a}, \bar{\boldsymbol{\alpha}} \rangle + \omega_{p+1}(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_{p+1}\} \\
& = & \langle \widetilde{\boldsymbol{a}}, \bar{\boldsymbol{\alpha}} \rangle + \omega_{p+1}(\widetilde{\boldsymbol{a}}) \text{ for some } \widetilde{\boldsymbol{a}} \in \mathcal{A}_{p+1}, \\
G_j & = & F_j^p \ (j \in K^p), \\
G_{p+1} & = & \emptyset \ \text{ or } \{\widetilde{\boldsymbol{a}}\}, \\
\boldsymbol{G} & = & (\boldsymbol{F}^p, G_{p+1}) \in \mathcal{F}^{\leq}(K^{p+1}), \\
B_{p+1} & = & \{\widetilde{\boldsymbol{a}}\}, \\
N_{p+1} & = & \{\boldsymbol{a} \in \mathcal{A}_{p+1} : \boldsymbol{a} \neq \widetilde{\boldsymbol{a}}\}, \\
\bar{x}_j(\boldsymbol{a}) & = & \begin{cases} \text{a generic positive number} & \text{if } \boldsymbol{a} \in B_j \text{ and } j \in K^{p+1}, \\ 0 & \text{if } \boldsymbol{a} \in N_j \text{ and } j \in K^{p+1}, \end{cases} \\
\boldsymbol{d} & = & \displaystyle\sum_{j \in K^{p+1}} \sum_{\boldsymbol{a} \in B_j} \boldsymbol{a}\ \bar{x}_j(\boldsymbol{a}), \\
b_j & = & \displaystyle\sum_{\boldsymbol{a} \in B_j} \bar{x}_j(\boldsymbol{a}) \ \ (j \in K^{p+1}).
\end{array}$$

By construction and Condition 4.1, we know that

- $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K^{p+1})$ is a basic feasible solution of $\mathrm{D}(K^{p+1}, \boldsymbol{G})$,
- $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$ is a basic feasible solution of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$,
- the two solutions satisfy the strict complementarity condition

$$\bar{x}_j(\boldsymbol{a}) = 0 \text{ if and only if } \omega_j(\boldsymbol{a}) + \langle \boldsymbol{a}, \bar{\alpha} \rangle - \bar{\beta}_j > 0 \quad (\boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1}).$$

Thus the two basic solutions constructed are optimal basic solutions of $\mathrm{D}(K^{p+1}, \boldsymbol{G})$ and $\mathrm{P}(K^{p+1}, \boldsymbol{G})$, respectively.

Since the new primal basic feasible solution $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$ of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$ shares its basic variables with the basic solution of $\mathrm{P}(K^p, \boldsymbol{F}^p)$ corresponding to the previous basic feasible solution $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K^p)$ of its dual $\mathrm{D}(K^p, \boldsymbol{F}^p)$ except one new variable $x_{p+1}(\widetilde{\boldsymbol{a}})$, we can easily recover the optimal simplex tableau or dictionary of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$ and its dual $\mathrm{D}(K^{p+1}, \boldsymbol{G})$. It should be also noted that all basic variables of the primal optimal solution $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$ of $\mathrm{P}(K^{p+1}, \boldsymbol{G})$ are positive. This implies that $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$ serves as a basic feasible solution of $\mathrm{P}(K^{p+1}, \boldsymbol{F})$ with any $\boldsymbol{F} = (\boldsymbol{F}^p, F_{p+1}) \in \mathcal{F}^{\leq}(K^{p+1})$. Therefore, for any $\boldsymbol{F} = (\boldsymbol{F}^p, F_{p+1}) \in \mathcal{F}^{\leq}(K^{p+1})$, we can perform the infeasibility test based on primal problems, which we have presented in Section 4.3, by applying the primal simplex pivot from $\bar{\boldsymbol{x}} = (\bar{x}_j(\boldsymbol{a}) : \boldsymbol{a} \in \mathcal{A}_j,\ j \in K^{p+1})$.

On the other hand, the dual optimal basic solution $(\bar{\boldsymbol{\alpha}}, \bar{\beta}_j : j \in K^{p+1})$ remains a basic feasible solution of $\mathrm{D}(K^{p+1}, \boldsymbol{G})$ even if we replace its objective function. Therefore we can also start the infeasibility and feasibility tests based on dual problems, which we have presented in Sections 4.1 and 4.2, respectively.

5 Implementation

In this section, we discuss implementation of our computational method using the infeasibility and feasibility tests based on primal problems for the fully mixed case where $m = n$, $M = \{1, 2, \ldots, n\}$ and $s_j = 1$ $(j = 1, 2, \ldots, n)$. In this case, the family of subsets $\mathcal{F}^{\leq}(K)$, $\mathcal{F}^{=}(K)$ and $\mathcal{F}^{=\&\mathrm{f}}(K)$ defined in (5) turn out to be

$$\begin{aligned}
\mathcal{F}^{\leq}(K) &= \{\boldsymbol{F} = (F_j : j \in K) \ : \ F_j \subset \mathcal{A}_j,\ \sharp F_j \leq 2\ (j \in K)\}, \\
\mathcal{F}^{=}(K) &= \{\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{\leq}(K) \ : \ \sharp F_j = 2\ (j \in K)\}, \\
\mathcal{F}^{=\&\mathrm{f}}(K) &= \{\boldsymbol{F} = (F_j : j \in K) \in \mathcal{F}^{=}(K) \ : \ \mathrm{E}(K, \boldsymbol{F}) \text{ is feasible}\},
\end{aligned}$$

respectively. We provide two algorithms; one is a serial algorithm on a single computer and the other is a parallel algorithm on a client-server based parallel computing system. In both algorithms, we take $\ell = m = n$ and $K^0, K^1, K^2, \ldots, K^n$ as in (4).

5.1 A serial enumeration algorithm on a single CPU

We fix some ordering 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.*,

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

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 $L(\mathcal{A}_j)$. For the convenience of the succeeding discussions on the parallel algorithm in the next subsection, we introduce two input parameters $r \in \{0, 1, 2, \ldots, n\}$ and $\boldsymbol{F}^r \in \mathcal{F}^=(K^r)$. In this subsection, we fix $r = 0$ and $\boldsymbol{F}^r = \emptyset \in \mathcal{F}^=(K^r) \equiv \{\emptyset\}$.

Algorithm 5.1. (Serial enumeration algorithm)

Step 0: If $\mathrm{E}(K^r, \boldsymbol{F}^r)$ is infeasible, then terminate. Let $p = r$.

Step 1: Letting $\boldsymbol{F}^{p+1} = (\boldsymbol{F}^p, \emptyset)$, compute a pair of optimal basic solutions of $\mathrm{P}(K^{p+1}, \boldsymbol{F}^{p+1})$ and $\mathrm{D}(K^{p+1}, \boldsymbol{F}^{p+1})$ (as we have described in the previous section if $p \geq 1$). Let $p = p + 1$ and $\widetilde{\mathcal{A}_p} = \mathcal{A}_p$. If $p = 1$ then go to Step 2 (because the rest of Step 1 is not effective at all usually). Otherwise check whether pivoting each nonbasic variable $x_p(\bar{\boldsymbol{a}})$ with $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}_p}$ into the basic variables leads to the unbounded objective value as described in Lemma 4.5; if so, let $\widetilde{\mathcal{A}_p} = \widetilde{\mathcal{A}_p} \backslash \{\bar{\boldsymbol{a}}\}$. Go to Step 2.

Step 2: Solve the linear program $\mathrm{P}(K^p, (\boldsymbol{F}^{p-1}, \{\bar{\boldsymbol{a}}\}))$ for every $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}_p}$ by applying the primal simplex method to $\mathrm{P}(K^p, (\boldsymbol{F}^{p-1}, \{\bar{\boldsymbol{a}}\}))$. If $\mathrm{P}(K^p, (\boldsymbol{F}^{p-1}, \{\bar{\boldsymbol{a}}\}))$ is unbounded for some $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}_p}$, 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 $F^p_p = \emptyset$ and $\boldsymbol{F}^p = (\boldsymbol{F}^{p-1}, F^p_p)$ and go to Step 3.

Step 3: If $p = r$ then terminate. Otherwise, let

$$\boldsymbol{F}^p = (\boldsymbol{F}^{p-1}, \mathrm{succ}(F^p_p, L(\widetilde{\mathcal{A}_p}))).$$

Step 4: If $F^p_p = \emptyset$, then let $p = p - 1$ and go to Step 3. Otherwise, go to Step 5.

Step 5: Solve the linear program $\mathrm{P}(K^p, \boldsymbol{F}^p)$ and its dual $\mathrm{D}(K^p, \boldsymbol{F}^p)$ by applying the primal simplex method to $\mathrm{P}(K^p, \boldsymbol{F}^p)$. If $\mathrm{P}(K^p, \boldsymbol{F}^p)$ is unbounded, go to Step 3. Otherwise go to Step 6.

Step 6: If $p = n$, then output the optimal solution of $\mathrm{D}(K^p, \boldsymbol{F}^p)$ and go to Step 3. Otherwise go to Step 1.

The latter half of Step 1 tries to detect elements $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}_p}$ such that primal problems $\mathrm{P}(K^p, (\boldsymbol{F}^{p-1}, \{\bar{\boldsymbol{a}}\}))$ with $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}_p}$ are unbounded. By eliminating such $\bar{\boldsymbol{a}}$ 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 Steps 2 and 5 decreases. At Step 2, we further make unboundedness checks for the remaining $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}_p}$. Although we need to solve $\sharp\widetilde{\mathcal{A}_p}$ additional linear programs, Step 2 leads to a significant reduction in the number of linear programs to be solved at Step 5.

Remark 5.2. We can effectively apply the parametric simplex method to linear programs $\mathrm{P}(K^p, (\boldsymbol{F}^{p-1}, \bar{\boldsymbol{a}}))$ with $\bar{\boldsymbol{a}} \in \widetilde{\mathcal{A}}$ at Step 2, and also to a linear program $\mathrm{P}(K^p, \boldsymbol{F}^p)$ at Step 5.

Remark 5.3. We can extend Algorithm 5.1 to the general semi-mixed case where $M = \{1, 2, \ldots, m\}$. But we need some modification; we take all possible combinations choosing $(s_j + 1)$ elements from $\mathcal{A}_j$ for $\forall j \in M$ and arrange them in $L(\mathcal{A}_j)$ with the lexicographical order.

5.2 A parallel enumeration algorithm on a client-server based parallel computing system

Now suppose that we have a client-server based computer system consisting of N processors. We first choose $r \in \{1, 2, \ldots, n\}$, and partition Problem 2.1 into $\sharp\mathcal{F}^{=}(K^r) = \frac{1}{2}\Pi_{i=1}^{r} m_i(m_i - 1)$ subproblems.

$$
\begin{array}{rcl}
\text{Problem}(K^r, \boldsymbol{F}) & : & \text{Find a solution } (\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathrm{R}^{n+m} \text{ satisfying} \\
 & & \quad \beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in F_j, \ j = 1, 2, \ldots, r), \\
 & & \quad \beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle \leq \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in \mathcal{A}_j, \ j \in M) \\
 & & \text{with exactly 2 equalities for each } j \in M
\end{array}
$$

$(\boldsymbol{F} \in \mathcal{F}^{=}(K^r))$. In this subproblem, 2 equalities

$$\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle = \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in F_j)$$

for $j = 1, 2, \ldots, r$ are already specified, so that each solution of the subproblem must satisfy

$$\beta_j - \langle \boldsymbol{a}, \boldsymbol{\alpha} \rangle < \omega_j(\boldsymbol{a}) \quad (\boldsymbol{a} \in \mathcal{A}_j \backslash F_j)$$

for $j = 1, 2, \ldots, r$ under Condition 2.2. We can apply Algorithm 5.1 with the input r and $\boldsymbol{F}^r = \boldsymbol{F} \in \mathcal{F}^{=}(K^r)$ to each Problem$(K^r, \boldsymbol{F})$ in parallel. It is reasonable to choose such an $r \in \{1, 2, \ldots, n\}$ that the number $\sharp\mathcal{F}^{=}(K^r)$ of subproblems generated is not less than the number N of processors. As we have pointed out at the end of Section 3, we also need to take the size of each subproblem into account for effective parallel computation.

Algorithm 5.4. (Parallel enumeration algorithm)

Step 0: Choose an $r \in \{1, 2, \ldots, n\}$ such that $N \leq \sharp\mathcal{F}^{=}(K^r)$. Let $\widetilde{\mathcal{F}} = \mathcal{F}^{=}(K^r)$.

Step 1: The client machine assigns each subproblem Problem$(K^r, \boldsymbol{F})$ with $\boldsymbol{F} \in \widetilde{\mathcal{F}}$ to an idle server machine, and delete $\boldsymbol{F}$ from $\widetilde{\mathcal{F}}$. This assignment continues until $\widetilde{\mathcal{F}}$ becomes empty.

Step 2: Each server machine to which the client machine has assigned Problem$(K^r, \boldsymbol{F})$ with $\boldsymbol{F} \in \widetilde{\mathcal{F}}$ executes Algorithm 5.1 with the input r and $\boldsymbol{F}^r = \boldsymbol{F}$.

Table 1: Computational CPU Time

problem	Algorithm 5.1		MVLP		Li&Li	
	CPU time	Mem.(MB)	CPU time	Mem.(MB)	CPU time	Mem.(MB)
cyclic-8	1.8s	2.5	41.1s	21	1.8s	1.0
cyclic-9	20.4s	2.7	7m 03.7s	21	11.7s	1.1
cyclic-10	3m 07.8s	2.7	1h 00m 23.0s	21	1m 42.9s	1.1
cyclic-11	36m 40.3s	2.9	9h 43m 54.0s	21	18m 06.0s	1.1
eco-9	7.7s	2.5	25.0s	21	9.2s	2.4
eco-10	48.3s	2.7	2m 04.3s	21	53.4s	2.4
eco-11	5m 03.9s	2.8	10m 12.7s	21	6m 28.3s	2.4
eco-12	31m 27.3s	2.9	1h 06m 23.3s	21	41m 34.5s	2.4

6 Numerical results

We consider two types of fully-mixed benchmark polynomial equation systems; cyclic n-roots problem [3]:

$$\begin{array}{l} x_1 + x_2 + \cdots + x_n = 0 \\ x_1x_2 + x_2x_3 + \cdots + x_nx_1 = 0 \\ x_1x_2x_3 + \cdots + x_nx_1x_2 = 0 \\ \cdots \\ x_1x_2x_3\cdots x_n - 1 = 0, \end{array}$$

and n-dimensional economics problem [13]:

$$\begin{array}{l} (x_1 + x_1x_2 + x_2x_3 + \cdots + x_{n-2}x_{n-1})x_n - 1 = 0 \\ (x_2 + x_1x_3 + \cdots + x_{n-3}x_{n-1})x_n - 2 = 0 \\ \cdots \\ (x_{n-2} + x_1x_{n-1})x_n - (n-2) = 0 \\ x_{n-1}x_n - (n-1) = 0 \\ x_1 + x_2 + \cdots + x_{n-1} + 1 = 0. \end{array}$$

Corresponding to these benchmark polynomial equation systems, we construct Problem 2.1 with real numbers $\omega_j(\boldsymbol{a})$ $(\forall \boldsymbol{a} \in \mathcal{A}_j, \forall j \in M = \{1, 2, \ldots, n\})$ randomly generated in the interval (0, 50).

To compare our method with some existing methods for mixed cell computation, we solved our test problems using a standard serial code based on Algorithm 5.1. The program was coded in C++ language and was executed on a DEC Alpha Workstation (CPU 21164 600MHz with 1GB memory). Table 1 shows the computational CPU time and the amount of memory required for cyclic n-roots problems (abbreviated by cyclic-n) and n-dimensional economics problems (abbreviated by eco-n), comparing Algorithm 5.1 with two existing software packages, the MVLP algorithm [5] and the Li&Li algorithm [12]. Note that Algorithm 5.1 outperforms MVLP algorithm in terms of CPU time and also required memory storage. There is, however, no significant difference between Algorithm 5.1 and the Li&Li algorithm in the CPU time and the used memory; Algorithm 5.1 is about two times slower than the Li&Li algorithm in cyclic n-roots problems, while the former algorithm is slightly faster than the latter in n-dimensional economic problems. We also observe that Algorithm 5.1 requires more memory storage than the Li&Li algorithm. This is because Algorithm 5.1 stores inverses of basis matrices of some linear programs to save the computation related to the inversion of basis matrices.

We implemented our parallel algorithm (Algorithm 5.4) on a Ninf (Network based Information Library for high performance computing) system, a global network-wide computing infrastructure which has been developed for high-performance numerical computation services. The Ninf system supports client-server based computing, as Figure 1 shows. It also provides a simple programming interface to utilize the computational resources on server machines across the wide area network. For further details on the Ninf system, the reader should refer to the articles [14, 15]. We executed Algorithm 5.4 on a PC cluster, which consists of 64 server machines with 128 processors. Each server machine on the PC cluster has 2 CPUs of Intel Pentium III 824MHz with 640MB memory. All server machines are locally connected via fast network services. The total performance of the PC cluster is about 52 Gflops (LINPACK). On the numerical experiments, we varied the number of processors N from 1 through 128 stepping by powers of 2.

Corresponding to the benchmark polynomial equation systems, Tables 2 and 3 show computational time on N parallel processors. In these tables, the empty entries show that the parallel algorithm requires either more than 5 hours or less than 1 minute to obtain all solutions of Problem 2.1. Recently, T.Y. Li and X. Li [12] solved cyclic 13-roots problem, which had been the largest cyclic n-roots problem solved, in 28h 3m 5s of computational CPU time on a 400MHz Intel Pentium II CPU with 256 MB memory. Algorithm 5.4 has renewed this record to cyclic 14-roots problem using 128 CPUs in less than 5 hours.

Figure 2 depicts how the computational real time decreases as the number of processors N increases for solving cyclic 12-roots problem. The horizontal line indicates the number of processors, and the vertical line the computational time in second, both in the log scale by 2. The parallel codes could solve cyclic 12-roots problem almost N times faster on N processors than on single processor. This result implies that the computational tasks required by our serial algorithm (Algorithm 5.1) are split to N processors in a highly parallel manner, with a little overhead.

7 Summary

In this paper, we have presented a computational method for allocating fine mixed cells of a polynomial equation system. Those cells play a crucial role in constructing polyhedral homotopies $\boldsymbol{h}(\boldsymbol{x}, t) = \boldsymbol{0}$ for computing all solutions of a polynomial system. Our method effectively utilizes the duality theory and the simplex method for linear programs. A single machine implementation (Algorithm 5.1) of our method works as efficiently as the state-of-art method given by Li-Li [12], and a parallel implementation solved a large scale problem, cyclic 14-roots problem using 128 CPUs in less than 5 hours for the first time.

Figure 1: Client-Sever Based Computing System

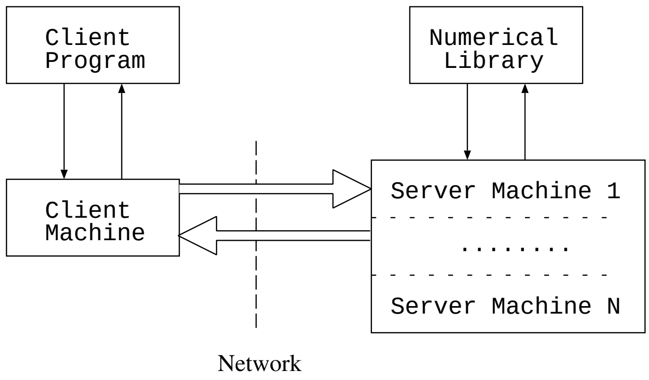


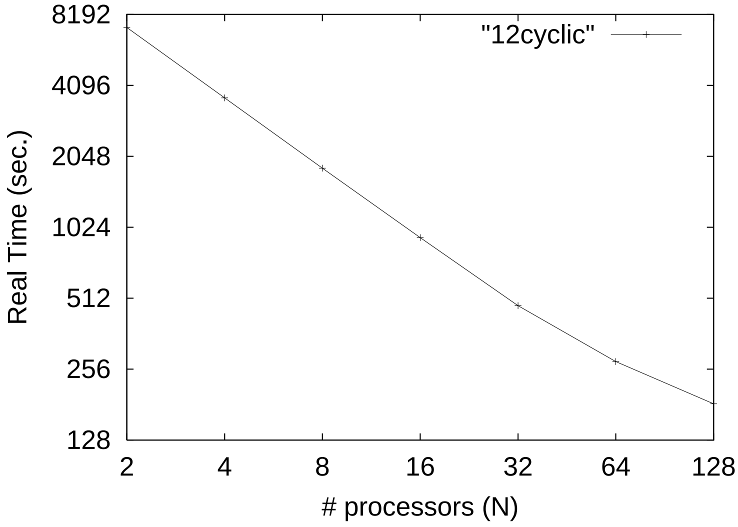
Table 2: Computational Real Time for cyclic n-roots problems

N	Cyclic-n Problems			
	cyclic11	cyclic12	cyclic13	cyclic14
1	27m 27s	4h 00m 03s		
2	13m 52s	2h 00m 14s		
4	6m 54s	1h 00m 24s		
8	3m 34s	30m 25s		
16	1m 47s	15m 26s	3h 15m 45s	
32	1m 07s	7m 56s	1h 38m 08s	
64		4m 36s	52m 35s	
128		3m 02s	30m 41s	4h 47m 22s

Table 3: Computational Real Time for eco-n problems

N	Eco-n Problems		
	eco12	eco13	eco14
1	22m 59s	2h 19m 59s	
2	11m 26s	1h 09m 60s	
4	5m 44s	35m 06s	
8	3m 01s	17m 44s	3h 28m 20s
16	1m 37s	9m 13s	1h 47m 51s
32	1m 06s	4m 47s	55m 39s
64		2m 57s	29m 39s
128		2m 53s	25m 23s

Figure 2: Effect on Computational Time by Increasing N



Acknowledgment: The authors would like to thank Professor T.Y. Li of Michigan State University. He introduced them the polyhedral homotopy method for computing all solutions of a polynomial system and the related problem of allocating all fine mixed cells. They also learned various issues on these subjects in detail when he visited them in July 2000. They also would like to thank Professor Satoshi Matsuoka of Tokyo Institute of Technology. He offered them to use an advanced PC cluster in his laboratory for their research.

References

[1] E. Allgower and K. Georg, *Numerical continuation methods,* Springer, New York, 1990.

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

[3] 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.

[4] S. N. Chow, J. Mallet-Paret and J. A. York, "A homotopy method for locating all zeros of a system of polynomials," in: *Functional differential equations and approximation of fixed points,* H. O. Peitgen and H. O. Walther editors, Lecture Notes in Mathematics **730** pp.77–88, Springer, New York, 1979.

[5] 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.

[6] T. Gao and T.Y. Li, "Mixed Volume Computation Via Linear Programming," Dept. of Mathematics, Michigan State University, East Lansing, MI 48824.

[7] C. B. Garcia and W. I. Zangwill, "Determining all solutions to certain systems of nonlinear equations," *Mathematics of Operations Research* **4** (1979) 1–14.

[8] C. B. Garcia and W. I. Zangwill, "Global calculation methods for finding all solutions to polynomial systems of equations in N variables," in: *Extremal Methods and System Analysis,* A. V. Fiacco and K. O. Kortanek editors, Lecture Notes in Economics and Mathematical Systems **174**, pp.481–497, Springer, New York, 1980.

[9] C. B. Garcia and W. I. Zangwill, *Pathways to Solutions, Fixed Points, and Equilibria,* Prentice-Hall, Eaglewood Cliffs, N.J., 1981.

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

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

[12] T.Y. Li and X. Li, "Finding Mixed Cells in the Mixed Volume Computation," Dept. of Mathematics, Michigan State University, East Lansing, MI 48824.

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

[14] 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: *High-Performance Computing and Networking*, P. Sloot and B. Hertzberger editors, Lecture Note in Computer Science **1225**, pp.491-502, Springer, New York, 1997.

[15] 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'96),* February 1996.

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

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