A Sublinear Parallel Algorithm for Stable Matching

Tomás Feder* Nimrod Megiddo† Serge A. Plotkin‡

Parallel algorithms for various versions of the stable matching problem are presented. The algorithms are based on the primal-dual interior path-following method for linear programming. The main result is that a stable matching can be found in $O^*(\sqrt{m})$ time by a polynomial number of processors, where m is the total length of preference lists of individuals.

1. Introduction

In this paper we consider networks made of gates of constant size. We focus of non-expansive networks (to be defined below). The problems of evaluating the gate to which a network converges, and of finding a stable configuration in a network, are quite simple in the context of sequential computation; they can all be solved in linear time in the scatter-free case (a special case, Mayr and Subramanian [5]), and in quadratic time in the general nonexpansive case (Feder [1]). An interesting question is the existence of sublinear parallel algorithms with a polynomial number of processors.

We present parallel algorithms for the above problems which run in $O^*(\sqrt{I})$ time, with a polynomial number of processors, where I is the size of the input and $f(I) = O^*(g(I))$ means there exists a constant k such that $f(I) \leq g(I)(\log I)^k$. Our approach is based on formulating the problems as linear programming problems and solving them with the primal-dual interior path-following method.

As an application, the problem of stable matching [3] can be solved in $O^*(\sqrt{m})$ time, where m is the total length of the preference lists of individuals. The approach is by means of interior point methods in linear programming.

In Sections 2 and 3 we introduce networks of gates and the concepts of nonexpansive and convergent networks. The material in these sections is from Feder [1]. In Section 4 we study the relation between these concepts and linear programming. In Section 5 we obtain the general result of recognizing stability in a network. This result is then applied to the stable matching problem in Section 6.

*IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120

†IBM Almaden Research Center, 650 Harry Road, San Jose, California 95120-6099, and School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel. Research supported in part by ONR contract N00014-91-C-0026.

‡Department of Computer Science, Stanford University, Stanford, CA 94305

2. Gates and Networks

A (boolean) *assignment* is a mapping $\boldsymbol{x} : S \rightarrow \{0,1\}$ with a domain $S = S(\boldsymbol{x})$. An element $i \in S(\boldsymbol{x})$ is a *coordinate* of $\boldsymbol{x}$, and the image $\boldsymbol{x}(i)$ is its *value*. Given a set of coordinates $T \subseteq S(\boldsymbol{x})$, we denote by $\boldsymbol{x}_T$ the restriction of $\boldsymbol{x}$ to the set T. If $T = \{i\}$ $(i \in S(\boldsymbol{x}))$ then $\boldsymbol{x}_T$ is denoted by x_i. If $\boldsymbol{x}$ and $\boldsymbol{y}$ are assignments with $S(\boldsymbol{x}) \cap S(\boldsymbol{y}) = \emptyset$, then $\boldsymbol{xy}$ denotes the *union* of the two assignments, with $S(\boldsymbol{xy}) = S(\boldsymbol{x}) \cup S(\boldsymbol{y})$. In particular, if $S(\boldsymbol{x}) = \{1,2,\ldots,n\}$, then $\boldsymbol{x} = x_1 x_2 \ldots x_n$. With a slight abuse of notation, we shall identify each x_i with its value $\boldsymbol{x}(i)$. For example, the statement $\boldsymbol{x} = x_1x_2x_3 = 011$ indicates that $S(\boldsymbol{x}) = \{1,2,3\}$ and $(\boldsymbol{x}(1), \boldsymbol{x}(2), \boldsymbol{x}(3)) = (0,1,1)$. Two assignments $\boldsymbol{x}$ and $\boldsymbol{y}$ are *consistent* if $x_i = y_i$ for all $i \in S(\boldsymbol{x}) \cap S(\boldsymbol{y})$.

A *gate* is a mapping $g : \{0,1\}^I \rightarrow \{0,1\}^O$ from assignments on the input set $I = I(g)$ to assignments on the output set $O = O(g)$. The coordinates in $I(g)$ and $O(g)$ are called *inputs* and *outputs* of g, respectively. The gate g is a k-input, ℓ-output gate if $|I(g)| = k$ and $|O(g)| = \ell$. Given a gate g, an assignment $\boldsymbol{x}$ with $S(\boldsymbol{x}) \subseteq I(g)$ and a coordinate set $T \subseteq O(g)$, the *restriction* $g_{\boldsymbol{x},T}$ of the gate g is the gate g' obtained from g by discarding the outputs not in T and discarding the inputs in $S(\boldsymbol{x})$ after assigning to them the values given by $\boldsymbol{x}$. More formally, the gate g' has inputs $I(g') = I(g) \setminus S(\boldsymbol{x})$, outputs $O(g') = O(g) \cap T$, and satisfies $g'(\boldsymbol{y}) = g(\boldsymbol{xy})_T$.

A *network* is a set of gates that share neither inputs nor outputs. This means that if N is a network and g and g' are distinct gates in N, then $I(g) \cap I(g') = \emptyset$ and $O(g) \cap O(g') = \emptyset$. On the other hand, given two (not necessarily distinct) gates g, g' in N, it may happen that an output of g is also an input of g'. If $i \in O(g) \cap I(g')$, then we say that output i of gate g and input i of gate g' are *linked*. By the disjointness property, every input is linked to at most one output, and every output is linked to at most one input. These links induce a topology on the network that can be described by a directed multigraph on the gates of the network, *i.e.*, a directed graph with the gates as the vertices, with a directed edge from g to g' for every output of g linked to an input of g'; loops and parallel edges are allowed. If the underlying directed multigraph of a network is acyclic, the network is called a *circuit*.

The *transition function* of a network N is a single gate f which is equivalent to the entire network as we explain below. The gate f has $I(f) = \bigcup_{g \in N} I(g)$ and $O(f) = \bigcup_{g \in N} O(g)$, and satisfies $\boldsymbol{y} = f(\boldsymbol{x})$ if and only if $\boldsymbol{y}_{O(g)} = g(\boldsymbol{x}_{I(g)})$ for all gates $g \in N$. Note that if f is the transition function of N, then the networks N and $N' = \{f\}$ have the same transition function; we shall see that, for many purposes, they can actually be treated as the same network. The set $R(N) = I(f) \cup O(f)$ of a network N with transition function f is the set of *coordinates* of the network N. It consists of three disjoint subsets: the set of *links* $L(N) = I(f) \cap O(f)$, the set of *inputs* $I(N) = I(f) \setminus O(f)$, and the set of *outputs* $O(N) = O(f) \setminus I(f)$ of the network.

A *configuration* of a network N is an assignment $\boldsymbol{u}$ on the coordinate set $R(N)$, and consists of an *input assignment* $\boldsymbol{u}_{I(N)}$, an *output assignment* $\boldsymbol{u}_{O(N)}$, and an *internal assignment* $\boldsymbol{u}_{L(N)}$. A network N can be used to define an associated mapping on the configurations of N. Given two configurations $\boldsymbol{x}$ and $\boldsymbol{y}$ of a network N with transition function f, we write $\boldsymbol{y} = N(\boldsymbol{x})$ if $\boldsymbol{y}_{I(N)} = \boldsymbol{x}_{I(N)}$ and $\boldsymbol{y}_{O(N) \cup L(N)} = f(\boldsymbol{x}_{I(N) \cup L(N)})$. In other words, all gates are evaluated using the values assigned to their inputs by $\boldsymbol{x}$ as inputs, thus obtaining at their outputs the values for the configuration $\boldsymbol{y}$; the inputs to the network are not outputs of any gate, and thus keep their value from $\boldsymbol{x}$. A configuration $\boldsymbol{x}$ is *stable* if $N(\boldsymbol{x}) = \boldsymbol{x}$. Thus, a configuration $\boldsymbol{x}$ is stable if it satisfies $f(\boldsymbol{x}_{I(f)}) = \boldsymbol{x}_{O(f)}$ for the transition function f or, equivalently, $g(\boldsymbol{x}_{I(g)}) = \boldsymbol{x}_{O(g)}$ for each gate $g \in N$, *i.e.*, if it satisfies all the gate equations.

The *k'th iterate* of a mapping τ on a set U is the mapping $\tau^{(k)}$ defined inductively by letting $\tau^{(0)}(z) = z$ and $\tau^{(k+1)}(z) = \tau(\tau^{(k)}(z))$ for all $z \in U$. A *periodic point* of τ is a z such that $\tau^{(p)}(z) = z$ for some $p \geq 1$. The least such p is the *period* of z. A *fixed point* of τ is a periodic point of period 1. We are particularly interested in the iterates and periodic points of the mapping associated with a network N. It will sometimes be useful to look at the iterates $N^{(k)}$ in terms of the transition function f of the network. For this purpose, we define two restrictions of f given an input assignment for the network. Given an assignment $\boldsymbol{x}$ on $I(N)$, the *output function* of the network is the mapping $g_{\boldsymbol{x}} = f_{\boldsymbol{x},O(N)}$, and the *internal function* of the network is the mapping $h_{\boldsymbol{x}} = f_{\boldsymbol{x},L(N)}$, so that if $\boldsymbol{z}$ is an assignment on $L(N)$, then $f(\boldsymbol{x}\boldsymbol{z}) = g_{\boldsymbol{x}}(\boldsymbol{z})h_{\boldsymbol{x}}(\boldsymbol{z})$. If $\boldsymbol{y}$ is an assignment on $O(N)$, then $N(\boldsymbol{x}\boldsymbol{y}\boldsymbol{z}) = \boldsymbol{x}g_{\boldsymbol{x}}(\boldsymbol{z})h_{\boldsymbol{x}}(\boldsymbol{z})$, and $N^{(k+1)}(\boldsymbol{x}\boldsymbol{y}\boldsymbol{z}) = \boldsymbol{x}\ g_{\boldsymbol{x}}(h^{(k)}_{\boldsymbol{x}}(\boldsymbol{z}))\ h^{(k+1)}_{\boldsymbol{x}}(\boldsymbol{z})$ for all $k \geq 0$. The periodic points of the mapping associated with N are called *periodic configurations*; the fixed points are precisely the stable configurations. The periodic configurations $\boldsymbol{x}\boldsymbol{y}\boldsymbol{z}$ consistent with an input assignment $\boldsymbol{x}$ are determined by the choice of a periodic point $\boldsymbol{z}$ of the internal function $h_{\boldsymbol{x}}$. For if $\boldsymbol{z}$ has period p and $\boldsymbol{z}' = h^{(p-1)}_{\boldsymbol{x}}(\boldsymbol{z})$, then the periodic configuration must have $\boldsymbol{z} = h_{\boldsymbol{x}}(\boldsymbol{z}')$ and $\boldsymbol{y} = g_{\boldsymbol{x}}(\boldsymbol{z}')$. Thus, the periodic configurations are the configurations $\boldsymbol{x}g_{\boldsymbol{x}}(z')h_{\boldsymbol{x}}(z')$ with $\boldsymbol{z}'$ a periodic point of $h_{\boldsymbol{x}}$.

3. Nonexpansive Mappings and Convergent Networks

The *distance* $d(\boldsymbol{x}, \boldsymbol{y})$ between two assignments $\boldsymbol{x}$ and $\boldsymbol{y}$ on a set S is defined by

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i \in S} |x_i - y_i| \ .$$

A gate g is *nonexpansive* if for any two assignments $\boldsymbol{x}$ and $\boldsymbol{y}$ on $I(g)$,

$$d(g(\boldsymbol{x}), g(\boldsymbol{y})) \leq d(\boldsymbol{x}, \boldsymbol{y}) \ .$$

A network N is said to be *convergent* if for every input assignment $\boldsymbol{x}$ there exists an output assignment $\boldsymbol{y}$ such that every configuration consistent with $\boldsymbol{x}$ maps to a configuration consistent with $\boldsymbol{y}$ under sufficiently many iterations of N. More precisely, for every configuration $\boldsymbol{u}$ consistent with $\boldsymbol{x}$, there exists an integer k_0 such that $N^{(k)}(\boldsymbol{u})$ is consistent with $\boldsymbol{y}$ for all $k \geq k_0$. Since every configuration maps to a periodic configuration for sufficiently large k, and every periodic configuration maps to itself for infinitely many values of k, the condition of convergence is equivalent to the requirement that every periodic configuration consistent with $\boldsymbol{x}$ must also be consistent with $\boldsymbol{y}$. Recall that the periodic configurations of N are the configurations $\boldsymbol{x}g_{\boldsymbol{x}}(\boldsymbol{z})h_{\boldsymbol{x}}(\boldsymbol{z})$, where $\boldsymbol{z}$ is a periodic point of $h_{\boldsymbol{x}}$ and the mappings $g_{\boldsymbol{x}}, h_{\boldsymbol{x}}$ are the output and internal functions of N (see Section 2). The condition defining convergent networks becomes then the statement that $g_{\boldsymbol{x}}(\boldsymbol{z}) = \boldsymbol{y}$ for all periodic points $\boldsymbol{z}$ of $h_{\boldsymbol{x}}$. If a network N is convergent, then for every input assignment $\boldsymbol{x}$ there is a *unique* corresponding output assignment $\boldsymbol{y}$ for N, and we say that N *converges* to the gate g with $I(g) = I(N)$ and $O(g) = O(N)$ that computes $g(\boldsymbol{x}) = \boldsymbol{y}$.

The notion of a convergent network evolved out of discussions between the first author and Ashok Subramanian, and was motivated by the following observation.

Lemma 3.1. *Every network of nonexpansive gates converge to a nonexpansive gate.*

Proof. Let f be the transition function of a network N of nonexpansive gates, given by $f(\boldsymbol{xz}) = g_{\boldsymbol{x}}(\boldsymbol{z})h_{\boldsymbol{x}}(\boldsymbol{z})$, where $g_{\boldsymbol{x}}$ and $h_{\boldsymbol{x}}$ are the output and internal functions. Let $\boldsymbol{z}$ and $\boldsymbol{z}'$ be periodic points of $h_{\boldsymbol{x}}$. Under these conditions,

$$\begin{aligned} d(g_{\boldsymbol{x}}(\boldsymbol{z}), g_{\boldsymbol{x}}(\boldsymbol{z}')) + d(h_{\boldsymbol{x}}(\boldsymbol{z}), h_{\boldsymbol{x}}(\boldsymbol{z}')) = \\ &= d(f(\boldsymbol{xz}), f(\boldsymbol{xz}')) \le d(\boldsymbol{xz}, \boldsymbol{xz}') \\ &= d(\boldsymbol{z}, \boldsymbol{z}') = d(h_{\boldsymbol{x}}(\boldsymbol{z}), h_{\boldsymbol{x}}(\boldsymbol{z}')) \ , \end{aligned}$$

so $d(g_{\boldsymbol{x}}(\boldsymbol{z}), g_{\boldsymbol{x}}(\boldsymbol{z}')) = 0$ and $g_{\boldsymbol{x}}(\boldsymbol{z}) = g_{\boldsymbol{x}}(\boldsymbol{z}')$. Therefore, the output $y = g_{\boldsymbol{x}}(\boldsymbol{z})$ depends only on $\boldsymbol{x}$, and not on the choice of a periodic point $\boldsymbol{z}$. This shows that the network is convergent, and converges to some gate g, where $g(\boldsymbol{x}) = g_{\boldsymbol{x}}(\boldsymbol{z})$ for all periodic points $\boldsymbol{z}$ of $h_{\boldsymbol{x}}$.

Given two input assignments $\boldsymbol{x}$ and $\boldsymbol{x}'$, let $\boldsymbol{z}$ and $\boldsymbol{z}'$ be periodic points of $h_{\boldsymbol{x}}$ and $h_{\boldsymbol{x}'}$, respectively, that are closest to each other. In particular, $d(\boldsymbol{z}, \boldsymbol{z}') \le d(h_{\boldsymbol{x}}(\boldsymbol{z}), h_{\boldsymbol{x}'}(\boldsymbol{z}'))$. Then

$$\begin{aligned} d(g_{\boldsymbol{x}}(\boldsymbol{z}),&g_{\boldsymbol{x}'}(\boldsymbol{z}')) + d(h_{\boldsymbol{x}}(\boldsymbol{z}), h_{\boldsymbol{x}'}(\boldsymbol{z}')) \\ &= d(f(\boldsymbol{xz}), f(\boldsymbol{x}'\boldsymbol{z}')) \le d(\boldsymbol{xz}, \boldsymbol{x}'\boldsymbol{z}') \\ &= d(\boldsymbol{x}, \boldsymbol{x}') + d(\boldsymbol{z}, \boldsymbol{z}') \\ &\le d(\boldsymbol{x}, \boldsymbol{x}') + d(h_{\boldsymbol{x}}(\boldsymbol{z}), h_{\boldsymbol{x}'}(\boldsymbol{z}')) \ . \end{aligned}$$

Thus, $d(g(\boldsymbol{x}), g(\boldsymbol{x}')) = d(g_{\boldsymbol{x}}(\boldsymbol{z}), g_{\boldsymbol{x}'}(\boldsymbol{z}')) \le d(\boldsymbol{x}, \boldsymbol{x}')$, so the gate g is nonexpansive.

4. Convergent Networks and Linear Programming

4.1 Linear characterizations of stability. Let N be a nonexpansive network with transition function f. A configuration $\boldsymbol{x}$ is stable if and only if for every configuration $\boldsymbol{a}$,

$$d(f(\boldsymbol{a}_{I(f)}), \boldsymbol{x}_{O(f)}) \le d(\boldsymbol{a}_{I(f)}, \boldsymbol{x}_{I(f)}) \ . \tag{4.1}$$

For every fixed $\boldsymbol{a}$, this is a linear inequality in terms of $\boldsymbol{x}$ since, for example,

$$d(\boldsymbol{a}, \boldsymbol{x}) = \sum_{i:a_i=0} x_i + \sum_{i:a_i=1} (1 - x_i) \ .$$

Definition 4.1. Denote by $\mathcal{A}$ the system of all the linear inequalities (4.1) corresponding to the configurations $\boldsymbol{a}$, the inequalities $0 \le x_i \le 1$, and the initial assigments $\boldsymbol{x}_{I(N)} = \boldsymbol{a}_{I(N)}$ for some fixed input assignment $\boldsymbol{a}_{I(N)}$.

Proposition 4.2. *The system $\mathcal{A}$ has a solution.*

Proof. The nonexpansive mapping f can be extended into a continuous nonexpansive mapping $\tilde{f}$ on the full hypercube $[0,1]^m$, by defining

$$\tilde{f}(\boldsymbol{x}) = \sum_{\boldsymbol{a}} \prod_{i \in I(f)} w(\boldsymbol{x}, \boldsymbol{a}, i) f(\boldsymbol{a}),$$

where the sum ranges over all $\{0,1\}$-configurations $\boldsymbol{a}$, and

$$w(\boldsymbol{x}, \boldsymbol{a}, i) = \begin{cases} x_i & \text{if } a_i = 1 \\ 1 - x_i & \text{otherwise} \ . \end{cases}$$

For two assignments $\boldsymbol{y}$ and $\boldsymbol{z}$ on $L(N)$, we say that $g(\boldsymbol{y}) = \boldsymbol{z}$ if $\tilde{f}(\boldsymbol{a}_{I(N)}\boldsymbol{y}) = \boldsymbol{zu}$ for some assignment $\boldsymbol{u}$ on $O(N)$. By Brouwer's theorem, g has a fixed point $\boldsymbol{y}$, *i.e.*, $g(\boldsymbol{y}) = \boldsymbol{y}$. Such a point $\boldsymbol{a}_{I(N)}\boldsymbol{yu}$ satisfies the conditions in $\mathcal{A}$ by nonexpansiveness of $\tilde{f}$.

Proposition 4.3. *Given an input* $\{0,1\}$ *-assignment $\boldsymbol{a}_{I(N)}$, the system $\mathcal{A}$ has a unique solution for those variables corresponding to the coordinates in $O(N)$ which coincides with the value of the gate to which the network converges.*

Proof. Let $\boldsymbol{x}$ be a solution of $\mathcal{A}$ as proven in Proposition 4.2. Let $\boldsymbol{z}$ be an integer periodic configuration closest to $\boldsymbol{x}$. We claim that, as in the proof of Lemma 3.1, where nonexpansiveness was used to establish convergence for networks, the conditions in $\mathcal{A}$ imply that the outputs of the network take the same values for both $\boldsymbol{x}$ and $\boldsymbol{z}$. Unfortunately, the size of $\mathcal{A}$ is *exponential* since there are 2^m choices for $\boldsymbol{a}$. However, we may consider the gates $g \in N$ separately, and require instead

$$d(g(\boldsymbol{b}), \boldsymbol{x}_{O(g)}) \le d(\boldsymbol{b}, \boldsymbol{x}_{I(g)}) \ , \tag{4.2}$$

where $\boldsymbol{b}$ ranges over the possible input assignments for g. When the gates are of constant size, this gives a number of constraints that is *linear* in the number of gates.

Definition 4.4. Denote the system of linear inequalities (4.2), $0 \le x_i \le 1$, and $\boldsymbol{x}_{I(N)} = \boldsymbol{a}_{I(N)}$ by $\mathcal{B}$.

4.2 The primal-dual path following method. Consider a linear program of the form

$$(P) \qquad \begin{aligned} \text{Minimize } \ & \boldsymbol{c}^T\boldsymbol{x} \\ \text{subject to } \ & \boldsymbol{Ax} = \boldsymbol{b} \\ & \boldsymbol{x} \ge \boldsymbol{0} \ . \end{aligned}$$

The dual of (P) is

$$(D) \qquad \begin{array}{rl} \text{Maximize} & \boldsymbol{b}^T\boldsymbol{y} \\ \text{subject to} & \boldsymbol{A}^T\boldsymbol{y}+\boldsymbol{s}=\boldsymbol{c} \\ & \boldsymbol{s}\geq \boldsymbol{0} \; . \end{array}$$

The *central path* of this primal-dual pair (P, D) (Megiddo [6]) consists of all the points $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{s})$ that satisify the constraints of (P) and (D) together with the equations

$$x_i s_i = \mu \qquad (i = 1, 2, \ldots)$$

where μ varies over the positive reals. The *duality gap* associated with $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{s})$ is given by

$$\boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{y} = \boldsymbol{s}^T\boldsymbol{x} \; .$$

Kojima, Mizuno and Yoshise [4] and Monteiro and Adler [7] developed good algorithms for tracing the primal-dual central path. They showed, in particular, that for any constant $\delta > 0$, given an initial triple $(\boldsymbol{x}^0, \boldsymbol{y}^0, \boldsymbol{s}^0)$ on the central path, the duality gap $\boldsymbol{s}^T\boldsymbol{x}$ can be reduced in $O(\sqrt{N}\log((\boldsymbol{s}^0)^T\boldsymbol{x}^0))$ iterations to at most δ.

4.3 Special form linear programs.

Definition 4.5. A linear programming problem is said to be of *special form* if it calls for minimizing ϕ, where ϕ is a sum of terms of one of the forms x_i or $1-x_i$ for variables x_i, subject to constraints $\psi \leq 0$, where ψ is a sum of terms of one of the forms x_i, $1 - x_i$, $-x_i$, $x_i - 1$. All variables are constrained also by $0 \leq x_i \leq 1$, which can also be viewed as constraints $\psi \leq 0$ when written as $-x_i \leq 0$ and $x_i - 1 \leq 0$. We assume that a minimum has value $\phi = 0$.

Theorem 4.6. *A linear program of the special form of size m can be put into an equivalent form such that for any constant $\delta > 0$, after $O(\sqrt{m}\log m)$ iterations, the duality gap of the current solution is at most δ.*

Proof. First, we ensure that in each ψ, the number of terms of one of the forms $-x_i$ or $x_i - 1$ is precisely one plus the number of terms of one of the forms x_i or $1-x_i$. The difference between these two numbers can be decreased by one by adding an "artificial" variable x_i to ψ with $0 \leq x_i \leq 1$, or increased by one by subtracting such an artificial variable from ψ, and adding x_i to the objective function ϕ. (We show below the equivalence to the original problem.)

Second, we ensure that if the coefficient of x_i in ϕ is c_i, then the sum of the coefficients of this x_i in all the terms ψ is $1 - c_i$. This sum can be decreased by one by adding a constraint $-x_i \leq 0$, or increased by one by adding a constraint $x_i - 1 \leq 0$.

Third, for each constraint $\psi \leq 0$, we introduce a "slack" variable $\alpha \geq 0$ and replace the constraint by $\psi + \alpha = 0$.

The optimum of the resulting problem is still equal to 0, so at the optimum all the artificial variables that were added to the objective function vanish.

Now the primal problem has a solution with all variables set to $\frac{1}{2}$ by the first condition enforced above. The dual problem has a solution with all $y_i = -1$ and all $s_i = 1$, by the second condition enforced above. This solution is on the central path. Starting the algorithm of [4] or [7] from this point, it takes $O(\sqrt{m}\log m)$ iterations to get to a point where the duality gap is at most δ. Note that the transformation increased the size of the linear program only by a constant factor, giving the stated bound.

4.4 Convergent network evaluation.

The linear programming problem associated with a nonexpansive network N is:

$$\begin{array}{rl} \text{Minimize} & d(\boldsymbol{a}_{I(N)}, \boldsymbol{x}_{I(N)}) \\ \text{subject to} & d(g(\boldsymbol{b}), \boldsymbol{x}_{O(g)}) - d(\boldsymbol{b}, \boldsymbol{x}_{I(g)}) \leq 0 \\ & 0 \leq x_i \leq 1 \; , \end{array}$$

where $\boldsymbol{a}_{I(N)}$ is the input assignment to the network, g ranges over gates and $\boldsymbol{b}$ ranges over input assignments. The linear program is thus of the special form as in Definition 4.5, so Theorem 4.6 can be applied.

The duality gap can be reduced below any constant $\delta > 0$. Let $\boldsymbol{a}$ be a periodic configuration consistent with the input assignment $\boldsymbol{a}_{I(N)}$ which is closest to $\boldsymbol{x}$. Thus,

$$d(f(\boldsymbol{a}_{I(f)})_{L(N)}, \boldsymbol{x}_{L(N)}) \geq d(\boldsymbol{a}_{L(N)}, \boldsymbol{x}_{L(N)}) \; .$$

On the other hand, when the linear program is put into the equivalent form, we obtain

$$d(f(\boldsymbol{a}_{I(f)}), \boldsymbol{x}_{O(f)}) - \sum y_i \leq d(\boldsymbol{a}_{I(f)}, \boldsymbol{x}_{I(f)}) \; ,$$

where the $-y_i$'s are the artificial variables that were added to the objective function. This can be rewritten as

$$\begin{aligned} & d(\boldsymbol{a}_{O(N)}, \boldsymbol{x}_{O(N)}) + d(f(\boldsymbol{a}_{I(f)})_{L(N)}, \boldsymbol{x}_{L(N)}) \\ & \quad \leq d(\boldsymbol{a}_{L(N)}, \boldsymbol{x}_{L(N)}) + d(\boldsymbol{a}_{I(N)}, \boldsymbol{x}_{I(N)}) + \sum y_i \end{aligned}$$

because $f(\boldsymbol{a}_{I(f)})_{O(N)} = \boldsymbol{a}_{O(N)}$. On the other hand,

$$d(\boldsymbol{a}_{I(N)}, \boldsymbol{x}_{I(N)}) + \sum y_i \leq \delta \; ,$$

since all the y_i's appear in the objective function, so by combining the three inequalities we get

$$d(\boldsymbol{a}_{O(N)}, \boldsymbol{x}_{O(N)}) \leq \delta \ .$$

Provided that $\delta < \frac{1}{2}$, this can be used to obtain the value of $\boldsymbol{a}_{O(N)}$, which is the output value produced by the gate to which the nonexpansive network converges.

It is observed in Goldberg, Plotkin, Shmoys, and Tardos [2] that one iteration of the interior point algorithm can be performed in $O(\log^2 N)$ time in the concurrent-read concurrent-write (CRCW) PRAM model with N^3 processors. Thus, we have the following:

Theorem 4.7. *For any fixed integer k, and for any nonexpansive network N with gates of at most k inputs and outputs, the gate to which N converges can be evaluated in $O(\sqrt{m}\log m)$ iterations, and an overall parallel time of $O(\sqrt{m}\log^3 m)$ on an m^3-processor CRCW PRAM.*

5. Convergent Networks and Stable Configurations

Let f be a nonexpansive gate with $I(f) = O(f) = T$. A *fixed point* of f is an assignment $\boldsymbol{a}$ on T such that $f(\boldsymbol{a}) = \boldsymbol{a}$.

For a subset $S \subseteq T$, we define the *projection* f_S to be the gate g with $I(g) = O(g) = S$ such that $g(\boldsymbol{a}) = \boldsymbol{b}$ if and only if for every periodic point, $\boldsymbol{z}$, of $f_{\boldsymbol{a},T\setminus S}$, there exists a $\boldsymbol{z}'$ such that $f(\boldsymbol{az}) = \boldsymbol{bz}'$. Thus, if f is the transition function of a network N, then f_S can be defined as the gate to which the network N converges. The following is from Feder [1].

Lemma 5.1.

(i) *A nonexpansive mapping f has a fixed point if and only if for every $S \subseteq T$ with $|S| = 1$, f_S has a fixed point.*

(ii) *A configuration $\boldsymbol{a}$ is a fixed point of f if and only if for all $S \subseteq T$ with $|S| = 2$, $\boldsymbol{a}_S$ is a fixed point of f_S.*

(iii) *In (ii), if f is the transition function of a network N, then S may be restricted to sets of two elements that are inputs or outputs of the same gate.*

Corollary 5.2. *The set of fixed points of f can be characterized as an instance of the 2-satisfiability problem with clauses $(x_i \neq a_i) \vee (x_j \neq a_j)$ for all i, j, a_i, and a_j such that $a_i a_j$ is not a fixed point of $f_{\{i,j\}}$. If f is the transition function of a network N, then $\{i, j\}$ may be restricted to two elements that are inputs or outputs of the same gate.*

It follows that the question of deciding whether a nonexpansive network has a stable configuration reduces to $2m$ evaluations of gates to which nonexpansive networks converge, and the search for a stable configuration reduces to $4\binom{m}{2}$ evaluations of such gates; in fact only $O(m)$ evaluations are needed here if for some fixed integer k the gates have at most k inputs and outputs. Since 2-SATISFIABILITY is in the class, we obtain:

Theorem 5.3. *For any fixed integer k, there exists an m^4-processor $O(\sqrt{m}\log^3 m)$ time CRCW PRAM algorithm that finds a stable configuration in a nonexpansive network with gates of at most k inputs and outputs.*

6. Network Stability and Stable Matching

Definition 6.1. The *X-gate* is a 2-input, 2-output gate which on inputs x_1, x_2 produces outputs y_1, y_2, such that

$$(y_1, y_2) = \begin{cases} (0,0) & \text{if } (x_1, x_2) = (1,1) \\ (x_1, x_2) & \text{otherwise} \ . \end{cases}$$

It is easy to see that the X-gate is nonexpansive. Subramanian [8] showed that the stable matching problem can be viewed as the problem of finding a stable configuration of a network of X-gates. The coordinates of the network are pairs ij, where i is the name of an individual and $0 \leq j \leq \ell_i$, where ℓ_i is the length of the preference list of individual i. If the jth choice of individual i is individual i', and the j'th choice of individual i' is individual i, then the network has an X-gate with the coordinates $i(j-1)$ and $i'(j'-1)$ as inputs, and the coordinates ij and $i'j'$ as outputs. The input $i0$ has the value $x_{i0} = 1$. Thus, in a stable configuration, the values x_{ij} for a fixed individual i are monotonically nonincreasing. If there is an index j such that $x_{i(j-1)} = 1$ and $x_{ij} = 0$, then i is matched to its jth choice. There can be at most one such index. The outputs $i\ell_i$ indicate whether i is matched to some partner, and are independent of the choice of stable matching.

It is of interest to see what the conditions defined by the linear program correspond to in the case of X-gates and comparators.

Proposition 6.2. *Two pairs $(x_1, x_2), (y_1, y_2) \in \{0,1\}^2$ satisfy the X-gate relation $y_1 y_2 = X(x_1 x_2)$ if and only if they satisfy the following linear system:*

$$(X) \qquad \begin{aligned} y_1 &= x_1 - \Delta \\ y_2 &= x_2 - \Delta \\ \text{with } \ \Delta &\geq \max(0, x_1 + x_2 - 1) \ . \end{aligned}$$

Proof. Recall that

$$X(00) = X(11) = 00,\ X(01) = 01,\ X(10) = 10\ .$$

For $\boldsymbol{a} = 10$, $X(\boldsymbol{a}) = 10$ and the nonexpansiveness condition is

$$(1 - y_1) + y_2 \le (1 - x_1) + x_2\ ,$$

whereas for $\boldsymbol{a} = 01$, $X(\boldsymbol{a}) = 01$ and the nonexpansiveness condition is

$$y_1 + (1 - y_2) \le x_1 + (1 - x_2)\ .$$

Thus,

$$\Delta \equiv x_1 - y_1 = x_2 - y_2\ .$$

On the other hand, for $\boldsymbol{a} = 00$, $X(\boldsymbol{a}) = 00$ and the nonexpansiveness condition is

$$y_1 + y_2 \le x_1 + x_2\ ,$$

which is equivalent to $\Delta \ge 0$, whereas for $\boldsymbol{a} = 11$, $X(\boldsymbol{a}) = 00$ and the nonexpansiveness condition is

$$y_1 + y_2 \le (1 - x_1) + (1 - x_2)\ ,$$

which is equivalent to

$$(x_1 - y_1) + (x_2 - y_2) \ge 2x_1 + 2x_2 - 2\ ,$$

or

$$\Delta\ \ge\ x_1 + x_2 - 1\ .$$

From the results in the last two sections, we obtain:

Theorem 6.3.

(i) *For n individuals with preference lists (over the set of individuals) of total length m, the set of people that are matched in stable matchings can be found in $O(\sqrt{m}\log^3 m)$ time on an m^3-processor CRCW PRAM.*

(ii) *If a stable matching exists, then it can be found in $O(\sqrt{m}\log^3 m)$ time on an m^4-processor CRCW PRAM.*

(iii) *A characterization of all the stable matchings by means of a 2-satisfiability instance can be found within the bounds in (ii).*

References

[1] T. Feder, *Stable Networks and Product Graphs*, doctoral dissertation, Stanford University (1991).

[2] A. V. Goldberg, S. A. Plotkin, D. B. Shmoys, and É. Tardos, "Using interior-point methods for fast parallel algorithms for bipartite matching and related problems," *SIAM J. Comput.* **21** (1992) 140–150.

[3] D. Gusfield and R. W. Irving, *The Stable Marriage Problem: Structure and Algorithms,*, MIT Press Series in the Foundations of Computing, MIT Press, Cambridge, MA, 1989.

[4] M. Kojima, S. Mizuno, and A. Yoshise, "A polynomial–time algorithm for a class of linear complementarity problems," *Mathematical Programming* **44** (1989) 1–26.

[5] E. Mayr and A. Subramanian, "The complexity of circuit value and network stability," *J. Comp. Syst. Sci.* **44** (1992), 302–323.

[6] N. Megiddo, "Pathways to the optimal set in linear programming," in: *Progress in Mathematical Programming: Interior-Point and Related Methods*, N. Megiddo, ed., Springer-Verlag, New York, 1988, pp. 131–158.

[7] R. D. C. Monteiro and I. Adler, "Interior path following primal-dual algorithms, Part I: Linear programming," *Mathematical Programming* **44** (1989) 27–41.

[8] A. Subramanian, "A new approach to stable matching problems," Technical Report STAN-CS-89-1275, Dept. of Computer Science, Stanford University, 1989.