

# A Fast Distributed Algorithm for Approximating the Maximum Matching<sup>\*</sup>

Andrzej Czygrinow<sup>1</sup>, Michał Hańcówkiak<sup>2</sup>, and Edyta Szymańska<sup>2</sup>

<sup>1</sup> Department of Mathematics and Statistics  
Arizona State University  
Tempe, AZ 85287-1804, USA  
andrzej@math.la.asu.edu

<sup>2</sup> Faculty of Mathematics and Computer Science  
Adam Mickiewicz University  
Poznań, Poland  
mhanckow@amu.edu.pl  
edka@amu.edu.pl

**Abstract.** We present a distributed approximation algorithm that computes in every graph  $G$  a matching  $M$  of size at least  $\frac{2}{3}\beta(G)$ , where  $\beta(G)$  is the size of a maximum matching in  $G$ . The algorithm runs in  $O(\log^4 |V(G)|)$  rounds in the synchronous, message passing model of computation and matches the best known asymptotic complexity for computing a maximal matching in the same protocol. This improves the running time of an algorithm proposed recently by the authors in [2].

## 1 Introduction

Any set of pairwise disjoint edges constitutes a matching of a given graph. From the algorithmic point of view two basic problems arise in the context of matchings. One is to compute any matching of maximum cardinality (maximum matching), the second one is to construct a matching not contained in any other matching of a given graph (inclusion maximal matching). There is rich literature concerning sequential algorithms for the maximum matching problem (see Lovász and Plummer [5] for an excellent survey). Several sequential algorithms for general graphs which are polynomial in time are known and many of them use augmenting paths as a main tool. It has turned out, however, that even the very simple sequential greedy algorithm is not known to be implemented in parallel. Many problems become even more difficult when the parallel network is not equipped with the shared memory storage. The increasing importance of large-scale networks such as Internet, ad-hoc and sensor networks have motivated active research on distributed computing in a message passing systems. Adopting the existing sequential procedures results in a rather very inefficient scheme and a new algorithmic approach is needed.

We consider the distributed model of computation introduced by Linial in [6]. In this model a network is represented by an undirected graph where each

---

<sup>\*</sup> Research supported by KBN grant no. 7 T11C 032 20

vertex stands for a processor and each edge corresponds to a connection between two processors in the network. Each processor has a unique ID and knows the number of vertices in the graph but not the global topology of the network. We assume full synchronization of the network: computation is performed in steps. In a single step, each processor can send a message to all of its neighbors, collect messages from its neighbors, and perform some local computations. Note that the above model is completely different from the parallel PRAM model where each computational unit can communicate with all other units at each step of the computations. As a result many problems that admit efficient algorithms in the PRAM model are still open in the distributed model. An eminent example, which is a generalization of the problem studied in this work, is the Maximal Independent Set (MIS) Problem. Efficient  $\mathcal{NC}$  algorithms for MIS are well-known ([7]) but the question if there is an efficient distributed algorithm for the problem is one of the main open problems in the area ([6]). However, an efficient distributed algorithm for the related Maximal Matching (MM) Problem is known due to Hańkowiak, Karoński, and Panconesi. In [4] the three authors proved the following theorem.

**Theorem 1 ([4]).** *There is a distributed procedure MATCH which in  $O(\log^4 n)$  many communication rounds in the synchronous, distributed model of computation, computes a maximal matching  $M$  in any graph  $G$  on  $n$  vertices.*

The procedure MATCH is a starting point for further research in [2] and in this paper. In [2], building on ideas from [4], the authors designed an algorithm which finds a maximal matching of size which is relatively close to the maximum:

**Theorem 2 ([2]).** *There is a distributed algorithm which in  $O(\log^6 n)$  steps finds a matching  $M$  in any graph  $G$  on  $n$  vertices, such that*

$$|M| \geq \frac{2}{3}\beta(G)$$

where  $\beta(G)$  denotes the size of the largest matching in  $G$ .

In this paper, we propose a substantial modification of the algorithm from Theorem 2 which runs in  $O(\log^4 n)$  steps and therefore has the same time complexity as the algorithm MATCH in Theorem 1. The following theorem is proved:

**Theorem 3.** *There is a distributed procedure MATCHING which in  $O(\log^4 n)$  steps finds in any graph  $G(V, E)$  with  $n = |V(G)|$  vertices a matching  $M$ , such that*

$$|M| \geq \frac{2}{3}\beta(G).$$

Although the approximation ratio of our result may not be very appealing, it should be remembered that the problem of computing an optimal solution in a general graph is even not known to be in  $\mathcal{NC}$ . And yet the approximation algorithm given by [3] is inherently designed for the EREW PRAM.

Interestingly, the result is purely deterministic and together with [4] stands among few examples of distributed computing, where polylogarithmic time complexity is achieved without the use of random bits. The  $O(\log^6 n)$  complexity of the algorithm from Theorem 2 comes from sequential iterations over  $O(\log^2 n)$  blocks (defined in the next section). It is this part of the algorithm which we improve. The modification that we present allows us to perform computations in parallel and as a result we drop the  $O(\log^2 n)$  factor in the time complexity altogether.

Following the strategy from [2], we compute a maximal matching  $M$  and then a maximal set of vertex-disjoint paths of length three that augment  $M$ . After augmenting the matching  $M$  along these paths we obtain a new matching  $M'$  such that  $|M'| \geq \frac{2}{3}\beta(G)$ . To find a maximal matching we invoke the procedure MATCH from Theorem 1 and to compute a maximal set of disjoint paths we design a new procedure, which is the main contribution of this paper.

In the next section, we present definitions and notation which are necessary to develop the algorithm together with some preliminary lemmas. The last section contains the description of the algorithm and a sketch of the proof of its correctness. For complete proofs the reader is referred to the full version of this paper.

## 2 Definitions, Notation and Preliminary Lemmas

In this section we fix some notation, introduce terminology, and state a few useful lemmas. As our algorithm creates an auxiliary multigraph, we define most of the terms in the multigraph setting.

Let  $M$  be a matching in a (multi)graph  $G = (V, E)$ . We say that a vertex  $v$  is  $M$ -saturated if  $v$  is an endpoint of some edge from  $M$ . An edge  $e = \{u, v\}$  is  $M$ -saturated if either  $u$  or  $v$  is  $M$ -saturated. A path  $P$  is  $M$ -alternating if it contains alternately edges from  $M$  and from  $E \setminus M$ . A path  $P$  of length  $2k + 1$ ,  $k \geq 0$ , augments  $M$  if  $P$  is  $M$ -alternating and no end of  $P$  is  $M$ -saturated. A special role in the paper will be played by paths of length three augmenting  $M$ . A path is called an  $(M, 3)$ -path if it augments  $M$  and has length three. A set of paths is called *independent* if every two paths from the set are vertex-disjoint.

Just like in [2], our algorithm is based on the following facts that are standard in graph theory.

### Claim 4

- (a) *Let  $M$  be an arbitrary matching in  $G$ . If there are no paths of length at most three augmenting  $M$ , then*

$$|M| \geq \frac{2}{3}\beta(G).$$

- (b) *Let  $M$  be a maximal matching in a graph  $G$  and let  $\mathcal{P}$  be a maximal independent set of paths of length three augmenting  $M$ . Further, let  $M' = M \div (\bigcup_{P \in \mathcal{P}} E(P))$  be the matching obtained by augmenting  $M$  along the paths from  $\mathcal{P}$ . Then there are no paths of length at most three augmenting  $M'$  in  $G$ .*

Consequently, parts (a) and (b) of Claim 4 together assure, that the matching  $M'$  satisfies

$$|M'| \geq \frac{2}{3}\beta(G).$$

In the process of constructing the matching we will use the notion of a *substantial* matching.

**Definition 1.** Let  $M$  be a matching in a multigraph  $G = (V, E)$  and  $\gamma > 0$ . A matching  $M$  is  $\gamma$ -substantial in  $G$  if the number of  $M$ -saturated edges of  $G$  is at least  $\gamma|E|$ .

Using an algorithm from [4] (with minor changes), we can find a substantial matching in a bipartite multigraph.

**Lemma 1.** For all constants  $C > 0$  and  $0 < \gamma < 1$  there exists a distributed algorithm that finds in  $O(\log^3 n)$  steps a  $\gamma$ -substantial matching in a bipartite multigraph  $G = (L, R, F)$ , where  $|F| \leq n^C$  and  $|L| + |R| = n$ .

Similarly, we will also use the notion of a substantial set of paths. Let  $\mathcal{P}_3(M)$  denote the set of all  $(M, 3)$ -paths in a graph  $G = (V, E)$ .

**Definition 2.** Let  $M$  be a matching in  $G$  and  $\gamma > 0$ . A set  $\mathcal{P}$  of  $(M, 3)$ -paths is called  $\gamma$ -path-substantial in  $G$  if the number of paths from  $\mathcal{P}_3(M)$  that have a common vertex with some path in  $\mathcal{P}$  is at least  $\gamma|\mathcal{P}_3(M)|$ .

Now we will introduce more technical terminology. Given a bipartite multigraph  $H = (L, R, E)$  we partition its left-hand side  $L$  into sets  $L_i = \{u \in L : \frac{D_i}{2} < \deg_H(u) \leq D_i\}$  for  $D_i = 2^i$  and  $i \geq 0$ . The submultigraph of  $H$  induced by the edges incident to  $L_i$ , denoted by  $H_i = (L_i, N(L_i), E_i)$  is called a  $D_i$ -block. A key concept which will be used in our approach is the concept of a spanner.

**Definition 3.** Let  $H = (A, B, E)$  be a  $D$ -block for some  $D$  as defined above. An  $(\alpha, K)$ -spanner from  $A$  to  $B$  is a subgraph  $S = (A', B, E')$  of  $H$  such that the following conditions are satisfied.

1.  $|A'| \geq \alpha|A|$ .
2. For every vertex  $a \in A'$ ,  $\deg_S(a) = 1$ .
3. For every vertex  $b \in B$ ,  $\deg_S(b) < \frac{K}{D}\deg_H(b) + 1$ .

In other words, a spanner is a collection of stars such that the degrees of the centers of the stars are appropriately bounded in terms of their degrees in  $H$ .

Note that spanners played an important role in designing the algorithm for maximal matching in [4]. Although the procedures in [4] are formulated for simple graphs they can be adopted to multigraphs with only minor changes. In particular, we have the following fact.

**Lemma 2 ([4]).** For every constant  $C$  and input parameter  $D$  there exist a constant  $K = K(C)$  and a distributed algorithm that finds in  $O(\log^3 n)$  steps a  $(\frac{1}{2}, K)$ -spanner in a multigraph  $H = (A, B, E)$  that is a  $D$ -block with  $|E| \leq n^C$  and  $n = |A| + |B|$ .

Specifically, in our case  $C = 4$  and  $K < 100$ .

In order to motivate the following definitions, we spend some time on explaining the main idea of our algorithm.

The approach used to find a set of augmenting paths is based on the following strategy. First we find a maximal matching  $M$  using the procedure MATCH given in Theorem 1. Then the maximal independent set of  $(M, 3)$ -paths is gradually constructed in rounds. In each round a substantial set of  $(M, 3)$ -paths is found. It turns out that we can find a substantial set of  $(M, 3)$ -paths in a special layered graph. For that reason, from the input graph, we obtain a virtual auxiliary graph which is of the form. Finally, once the paths are found in the layered graph, we translate them back to the original graph.

The layered graph will be a simple graph which has four layers of vertices and we will refer to it as a *4L-graph*. A precise construction of a 4L-graph is given in procedure REDUCE placed in the next section and here we just introduce its structural features. Every 4L-graph  $H = H(G, M)$ , with respect to  $G$  and a maximal matching  $M$ , has the vertex set  $V(H)$  partitioned into four nonempty sets  $X_1, X_2, X_3, X_4$  so that  $H[X_2, X_3] = M$  and every vertex from  $X_1$  is adjacent only to vertices from  $X_2$  and every vertex from  $X_4$  is adjacent only to vertices from  $X_3$ . In other words,  $H$  consists of 3 bipartite graphs. The edges of  $H[X_1, X_2]$  and  $H[X_3, X_4]$  are potential ingredients of the  $(M, 3)$ -paths extending  $M$ . The desired structure of a 4L-graph is obtained by first removing from  $E(G)$  all edges which violate the above properties. We will distinguish in the input graph  $G$  three types of edges with respect to  $M$ :

- (a) edges that are in  $M$ ,
- (b) edges that have exactly one endpoint in  $M$  (these edges may form triangles based on an edge from  $M$ ),
- (c) edges that are not in  $M$  but have both endpoints in  $M$ .

Note that edges from (c) do not belong to any  $(M, 3)$ -path of  $G$  and therefore we can delete them. Next, the triangles built of edges of type (b) are carefully destroyed (see procedure REDUCE). Finally, some vertices from  $V(G) \setminus V(M)$  are replicated to have their counterparts in  $X_4$  and  $X_1$ , respectively.

Let  $H = (X_1, X_2, X_3, X_4, E)$  be a 4L-graph. Now we introduce an auxiliary multigraph  $Mul(H) = (X_1, X_2, X_3, X_4, F)$  defined as follows.

**Definition 4.** Let  $H = (X_1, X_2, X_3, X_4, E)$  be a 4L-graph and for every  $e \in H[X_1, X_2] \cup H[X_3, X_4]$ , let  $w_e$  denote the number of  $(M, 3)$ -paths in  $H$  containing  $e$ . The multigraph  $Mul(H) = (X_1, X_2, X_3, X_4, F)$  is a graph obtained from  $H$  by putting  $w_e$  copies of  $e$  in  $F$  for every edge  $e \in H[X_1, X_2] \cup H[X_3, X_4]$ , and one copy of  $e$  for every edge  $e \in H[X_2, X_3]$  in  $F$ .

The main property of  $Mul(H)$  is that edges in  $Mul(H)[X_1, X_2]$  and similarly, in  $Mul(H)[X_3, X_4]$  correspond to  $(M, 3)$ -paths in  $H$ . In the course of our proof we will make use of it by finding a substantial matching in  $Mul(H)[X_1, X_2]$  and deducing a substantial property for a set of  $(M, 3)$ -paths built on the matching. Directly from Definition 4 we can derive the following fact.

**Fact 5** For every  $e = \{a, a'\} \in H[X_2, X_3]$ ,  $\deg_{Mul(H)}(a) = \deg_{Mul(H)}(a')$  which is equal to the number of  $(M, 3)$ -paths containing  $e$ . Moreover, there is a one-to-one correspondence between the edges in  $Mul(H)[X_1, X_2]$  and the  $(M, 3)$ -paths in  $H$ .

To this end, in  $Mul(H)$ , we define a  $4L$ - $D$ -block. The actual process of constructing the set of  $(M, 3)$ -paths will be performed in blocks that will partition the sets  $X_2$  and  $X_3$ .

**Definition 5.** Let  $R$  be the set of all edges  $e = \{a, a'\} \in Mul(H)[X_2, X_3]$  that satisfy

$$\frac{D}{2} < \deg_{Mul(H)}(a) = \deg_{Mul(H)}(a') \leq D$$

and let  $X'_2 = (\bigcup_{e \in R} e) \cap X_2$ ,  $X'_3 = (\bigcup_{e \in R} e) \cap X_3$ . A  $4L$ -sub-multigraph  $B_D(H) = (X_1, X'_2, X'_3, X_4, F')$  of  $Mul(H) = (X_1, X_2, X_3, X_4, F)$  which contains all edges incident to edges from  $R$  is called a  $4L$ - $D$ -block.

It is worth noticing here, that considering  $D = D(i) = 2^i$  for  $i = 0, 1, \dots, \log n$ ,  $4L$ -blocks  $B_{D(i)}$  are edge disjoint and partition the sets  $X_2$  and  $X_3$ . What is more, every subgraph  $B_{D(i)}(H)[X_3, X_4]$  is a  $D(i)$ -block.

### 3 Algorithm

In this section we present the main algorithm. In each iteration it constructs a substantial set of  $(M, 3)$ -paths and adds them to the global set of independent  $(M, 3)$ -paths. Then the paths are removed from the graph together with all edges incident to them. It stops when no such paths are left in the graph and outputs a new matching  $M^*$ . Below we present an outline of the algorithm to give some intuition behind our approach.

---

#### Procedure Matching

---

Input: Graph  $G$

Output: Matching  $M^*$

1. Find a maximal matching  $M$  in  $G$ .
2. For  $j := 1$  to  $O(\log n)$  do
  - (a) Construct a  $4L$ -graph  $\bar{G} = \bar{G}(G, M) = (X_1, X_2, X_3, X_4, E)$  from  $G$  and  $M$ .
  - (b) Construct a multigraph  $\tilde{G} = Mul(\bar{G}) = (X_1, X_2, X_3, X_4, F)$  with multiple edges in  $\bar{G}[X_1, X_2]$  and  $\bar{G}[X_3, X_4]$ .
  - (c) Find a  $\xi$ -path-substantial set  $\tilde{\mathcal{P}}$  of disjoint  $(M, 3)$ -paths in  $\tilde{G}$  using spanners in blocks in parallel.
  - (d) Translate  $\tilde{\mathcal{P}}$  found in  $\tilde{G}$  to  $\mathcal{P}$  in  $G$ .
  - (e) Accumulate the  $(M, 3)$ -paths from  $\mathcal{P}$ .
  - (f) Update  $G$  (remove from  $G$  the paths in  $\mathcal{P}$  with all edges incident to them).

3. Update  $M^*$  by augmenting  $M$  along the paths from  $\mathcal{P}$ .

---

The algorithm consists of a procedure that computes a substantial matching, and two procedures from [2]. First we describe a new procedure which computes a substantial set of  $(M, 3)$ -paths in a given 4L-graph. Then we recall two procedures from [2]; the first one reduces a graph  $G = (V, E)$  and a maximal matching  $M$  in  $G$  to a 4L-graph  $\bar{G} = (X_1, X_2, X_3, X_4, E)$ , the second one translates a  $\tilde{\alpha}$ -substantial set  $\tilde{\mathcal{P}}$  of independent  $(M, 3)$ -paths in  $\bar{G}$  to  $\alpha$ -substantial set  $\mathcal{P}$  of independent  $(M, 3)$ -paths in  $G$ .

### 3.1 Algorithm in a 4L-graph

In this section, we present the main part of our algorithm, PROCEDURE PATHS-IN4LGRAPH. This procedure finds a  $\xi$ -path-substantial set  $\mathcal{P}$  of disjoint  $(M, 3)$ -paths in a layered graph. We are building the paths in two phases. In the first phase we choose candidates for edges in  $\bar{G}[X_3, X_4]$  which will augment the matching  $M$ . This is performed by first constructing a star forest with all stars having centers in  $X_4$ . After the second phase at most one edge from every star is added to the set of  $(M, 3)$ -paths. Therefore we move along the star rays to the set  $X_2$  and introduce an auxiliary, bipartite graph with vertices corresponding to the stars we just created, on one side and  $X_1$  on the other. In the second phase we move our attention to this graph and find a substantial matching in it. After extending uniquely the matching to the graph  $\tilde{G}[X_3, X_4]$  we obtain the set  $\mathcal{P}$  of disjoint  $(M, 3)$ -paths.

Phase A: Steps 1-5 construct a star forest  $S'$  in  $H[X_3, X_4]$ .

Phase B: Steps 6-8 construct the set  $\mathcal{P}$  using  $S'$ .

Note that Phase B is analogous to the main algorithm for the 4L-graph in [2]. However, since the blocks in PATHSIN4LGRAPH are defined in a new way compared with [2], the analysis is different. The main problem in PATHSIN4LGRAPH is to design the star forest  $S'$  so that the stars can be glued together and used as super-vertices in Phase B. Properties of  $S'$  which we need to prove that the algorithm is indeed correct are quite delicate, on one hand we must delete some edges of a union of spanners  $S$  to make the analysis work, on the other we need to keep most of them to retain the substantial property. In particular, if a star has exactly one vertex in a block then it must have no vertices in any other block. This is taken care of in steps (4) and (5) of the procedure given below.

---

#### Procedure PathsIn4LGraph

---

Input: 4L-graph  $\bar{G} = (X_1, X_2, X_3, X_4, E)$ , where  $\bar{G}[X_2, X_3] = M$ .

Output: A set  $\mathcal{P}$  of disjoint  $(M, 3)$ -paths which is  $\xi$ -path-substantial in  $\bar{G}$  for some constant  $\xi > 0$ .

1. Construct the 4L-multigraph  $\tilde{G} := \text{Mul}(\bar{G})$  as in Definition 4.

2. For  $i = 0$  to  $4 \log n$  do  $D(i) := 2^i$   
 $B_i := B_{D(i)}(\tilde{G}) = (X_1, X_2(i), X_3(i), X_4, E_i)$ ,  
where  $X_j(i)$  is a subset of  $X_j$  corresponding to the block  $B_i$ , for  $j = 1, 2$ .
3. In parallel, for every  $i$ : Let  $K = K(4)$  be a constant in Lemma 2. Find a  $(\frac{1}{2}, K)$ -spanner  $S_i$  from  $X_3(i)$  to  $X_4$  in  $B_i[X_3(i), X_4]$  (using the procedure from [4]).
4. Let  $S = \bigcup_{i=0}^{4 \log n} S_i$  be a star forest.
  - (a) For any star  $Q \in S$  let  $Q_i := Q \cap S_i$ .
  - (b) For star  $Q$  in  $S$  let  $I_Q := \{i : |Q_i| = 1\}$  (i.e.  $Q_i$  is an edge) and let  $J_Q = \{i : |Q_i| > 1\}$ .
  - (c) Let  $i_Q$  be the number of edges of  $\tilde{G}[X_3, X_4]$  that are incident to vertices from  $\bigcup_{i \in I_Q} V(Q_i) \cap X_3$ . Let  $j_Q$  be the of edges of  $\tilde{G}[X_3, X_4]$  that are incident to vertices from  $\bigcup_{i \in J_Q} V(Q_i) \cap X_3$ .

$$i_Q = \sum_{\substack{v \in V(Q_i) \cap X_3 \\ i \in I_Q}} d_{\tilde{G}[X_3, X_4]}(v)$$

$$j_Q = \sum_{\substack{v \in V(Q_i) \cap X_3 \\ i \in J_Q}} d_{\tilde{G}[X_3, X_4]}(v)$$

5. In parallel, for every star  $Q \in S$ :
  - (a) If  $i_Q > j_Q$  then delete from  $Q$  all the edges that belong to  $\bigcup_{i \in J_Q} Q_i$ . In addition, select  $k \in I_Q$  such that  $k = \max I_Q$  and delete from  $Q$  all stars  $Q_i$  for which  $i \neq k$ . As a result  $Q$  is the edge  $Q_k$ .
  - (b) If  $i_Q \leq j_Q$  delete from  $Q$  all the edges that belong to  $\bigcup_{i \in I_Q} Q_i$ .
Let  $S'$  denote the star forest after these operations.
6. Construct the following auxiliary multi-graph  $\hat{G} = (X_1, X'_2, \hat{E})$  with  $X'_2 = \{N_{X_2}(Q') : Q' \in S'\}$  and the set of edges  $\hat{E} = \{\{u, v\} \in \hat{G} : u \in X_1 \text{ and } v \in N_{X_2}(Q') \text{ for some } Q' \in S'\}$ .
7. Find a  $1/2$ -substantial matching  $M'$  in  $\hat{G}$ .
8. Extend (in a unique way) every edge of  $M'$  to a path  $P$  of length three in the graph using an edge of matching  $M$  and an edge of a star in  $S'$ .  $\mathcal{P}$  is the set of all paths  $P$  obtained from  $M'$ .

---

We will need some more notation in the next two lemmas. Recall that  $S$  is a set of stars with centers in  $X_4$ . Let  $F_4$  be the set of vertices of stars from  $S$  which are in  $X_4$ ,  $F_3$  the set of vertices of stars in  $S$  which are in  $X_3$ . In addition let  $F_2 := N(F_3) \cap X_2$ . Similarly, by considering  $S'$ , we define  $F'_4, F'_3$ , and  $F'_2$ . The following lemma estimates the number of edges that we may lose while concentrating on the stars only.

**Lemma 3.** 1.  $e_{\tilde{G}}(F'_3, X_4) \geq \frac{1}{32} e_{\tilde{G}}(X_3, X_4)$   
2.  $e_{\tilde{G}}(X_1, F'_2) \geq \frac{1}{32} e_{\tilde{G}}(X_1, X_2)$ .

*Proof (Sketch).* By Fact 5, the proof of both parts is the same, and it follows from a property of the spanners  $S_i$  (see Part 1. of Definition 3).

**Lemma 4.** *Let  $K = K(4)$  be given in Lemma 2 and let  $\xi = \frac{1}{256K}$ . The set of paths  $\mathcal{P}$  found by PATHSIN4LGRAPH is  $\xi$ -path-substantial in  $G$ . The algorithm PATHSIN4LGRAPH runs in  $O(\log^3 V(\tilde{G}))$  steps.*

*Proof (Sketch).* First note that the time complexity comes from finding spanners in all the blocks (in parallel) in step (3) and by Lemma 2 is  $O(\log^3 n)$ . Let  $\mathcal{P}$  be the set of  $(M, 3)$ -paths found by the procedure PATHSIN4LGRAPH. In view of Fact 5, to prove that  $\mathcal{P}$  is  $\xi$ -path-substantial, we need to show that either  $\xi$ -fraction of edges in  $\tilde{G}[X_1, X_2]$  is incident to edges of paths from  $\mathcal{P}$  or  $\xi$ -fraction of edges in  $\tilde{G}[X_3, X_4]$  is incident to edges of paths from  $\mathcal{P}$ . For that we consider a matching  $M'$  in  $\tilde{G}$  found in Step (7) and the set  $W$  of vertices in  $X_2$  that are in  $M'$ -saturated supervertices and are not saturated by  $M'$  themselves. Then we consider two cases based on the number of edges incident to  $W$  with respect to  $e_{\tilde{G}}(X_1, X_2)$ . If it is relatively small then the fact that  $M'$  is  $\frac{1}{2}$ -substantial combined with Lemma 3 implies that  $\xi = \frac{1}{128}$ . The second case is a bit more complicated. We move our attention to the layers  $X_3$  and  $X_4$ . Incorporating the properties of the star forest  $S'$  and the upper bound on the degree of spanners, we show that at least  $\frac{1}{256}K$ -fraction of edges in  $E_{\tilde{G}}(X_3, X_4)$  is incident to centers of stars saturated by paths from  $\mathcal{P}$ . The two cases yield the Lemma with  $\xi = \frac{1}{256}K$ .

### 3.2 Reduction, Translation, and Modification

As indicated in previous sections our main algorithm can be summarized as follows. First compute a maximal matching  $M$ , next iterate  $O(\log n)$  times the following steps: (1) Reduce graph  $G$  and matching  $M$  to a layered graph  $\tilde{G}$ , (2) Find a set  $\tilde{P}$  of disjoint  $(M, 3)$ -paths in  $\tilde{G}$  using PATHSIN4LGRAPH, (3) Translate paths from  $\tilde{P}$  to paths  $P$  in  $G$  maintaining the substantial property, (4) Modify  $G$  and  $\tilde{G}$  with respect to  $P$ . In this section, we present procedures:

1. REDUCE which reduces  $G$  and  $M$  to a 4L-graph  $\tilde{G}$ ;
2. TRANSLATE which translates  $\tilde{P}$  to a set of paths  $P$  in  $G$ ;
3. MODIFY which modifies  $\tilde{G}$  with respect to  $P$ .

Procedures REDUCE and MODIFY are similar to procedures in [2], TRANSLATE differs from the corresponding procedure in [2] only by small details.

---

#### Procedure Reduce

---

Input: Graph  $G$  and a maximal matching  $M$  in  $G$ .

Output: 4L-graph  $\tilde{G}$ .

1. For  $e \in E(G)$  (in parallel) check if  $e$  is contained in at least one  $(M, 3)$ -path. If it is not then delete  $e$ . In particular, all edges from  $E(G) \setminus M$  which have both endpoints in  $M$  are deleted.
2. For every edge  $m = \{m_1, m_2\} \in M$ , with  $ID(m_1) < ID(m_2)$ , let  $T_m$  be the set of vertices in  $V$  such that every vertex from  $T_m$  is adjacent to  $m_1$  and  $m_2$ . Partition  $T_m$  into two groups  $T_{m,1}$  and  $T_{m,2}$  so that  $||T_{m,1}| - |T_{m,2}|| \leq 1$ .

- For every vertex  $t \in T_{m,1}$  delete the edge  $\{t, m_2\}$  from the graph, for every vertex  $t \in T_{m,2}$  delete  $\{t, m_1\}$ . As a result, the "new" graph  $G'$  does not have triangles based on edges from  $M$ . From now on we will operate on  $G'$ .
3. For every edge  $m = \{m_1, m_2\} \in M$ , with  $ID(m_1) < ID(m_2)$ , if  $e \in E(G')$  and  $e = \{v, m_1\}$  for some  $v \neq m_2$  then orient  $e$  from  $v$  to  $m_1$ , that is delete  $e$  and add arc  $(v, m_1)$ . If  $e \in E(G')$  and  $e = \{v, m_2\}$  for some  $v \neq m_1$  then orient  $e$  from  $m_2$  to  $v$ , that is delete  $e$  and add arc  $(m_2, v)$ .
  4. For every vertex  $v \in G \setminus V(M)$ , split  $v$  into two siblings  $v^-$  and  $v^+$ , where  $v^-$  inherits all the arcs that start in  $v$ ,  $v^+$  inherits arcs that end in  $v$ . Finally, ignore the orientation on the edges. As a result we obtain a 4L-graph  $\bar{G} = (V^-, V_1(M), V_2(M), V^+)$ , where  $V^- = \{v^-, v \in V(G) \setminus V(M)\}$ ,  $V^+ = \{v^+, v \in V(G) \setminus V(M)\}$ , and  $V_1(M), V_2(M)$  are corresponding endpoints of  $M$  that is if  $m = \{m_1, m_2\} \in M$  and  $ID(m_1) < ID(m_2)$  then  $m_1 \in V_1(M)$ ,  $m_2 \in V_2(M)$ .

The above procedure obtains  $\bar{G}$  by first deleting all triangles which are based on edges from  $M$  and then using orientation to split  $M$ -unsaturated vertices into  $V^-$  and  $V^+$  (see [2] for details). There are two main properties of REDUCE that are useful for our analysis.

**Lemma 5.** *Let  $\bar{m}_3$  denote the number of  $(M, 3)$ -paths in  $\bar{G}$ ,  $m'_3$  the number of  $(M, 3)$ -paths in  $G'$ , and  $m_3$  denote the number of  $(M, 3)$ -paths in  $G$ . Then*

1.  $m'_3 = \bar{m}_3$ ,
2.  $m_3/4 \leq m'_3$ .

**Fact 6** *If  $v_1^- m_1 m_2 v_2^+$  is an  $(M, 3)$ -path in  $\bar{G}$  then  $v_1 m_1 m_2 v_2$  is  $(M, 3)$ -path in  $G$ .*

Our next procedure modifies  $G$  with respect to the set of  $(M, 3)$ -paths  $\mathcal{P}$ . The procedure deletes all edges which are incident to paths from  $\mathcal{P}$ . As a result, it is possible that some edges present in the modified graph  $G'$  will not belong to any  $(M, 3)$ -path of  $G'$ . These edges are subsequently deleted in step (1) of REDUCE.

### Procedure Modify

Input: Graph  $G$  and a set  $\mathcal{P}$  of  $(M, 3)$ - paths in  $G$ .  
Output: Modified  $G'$ .

1. For any edge  $e \in \bigcup_{P \in \mathcal{P}} E(P)$  in parallel: delete edges that are incident to  $e$
2. Delete all edges from  $\bigcup_{P \in \mathcal{P}} E(P)$

Finally, in the last procedure of this section, we show how to translate a set of paths  $\bar{\mathcal{P}}$  in  $\bar{G}$  to a set of paths  $\mathcal{P}$  in  $G$ . Recall that  $G'$  denotes the subgraph of  $G$  obtained by destroying triangles that contain edges from  $M$ , in step (2) of REDUCE.

---

### Procedure Translate

---

Input: Set  $\bar{\mathcal{P}}$  of independent  $(M, 3)$ -paths in  $\bar{G}$ .

Output: Set  $\mathcal{P}$  of independent  $(M, 3)$ -paths in  $G$ .

1. Identify vertices  $v^-$  and  $v^+$  that correspond to one vertex  $v \in V(G)$ . As a result we obtain from  $\bar{\mathcal{P}}$  cycles and paths in graph  $G$ . These paths and cycles have lengths of the form  $3i$  where  $i > 1$  and are built up from  $(M, 3)$ -paths.
2. Treat  $(M, 3)$ -paths as edges between endpoints of these paths. Now the problem of selecting a set of independent  $(M, 3)$ -paths in  $G$  is reduced to the one of finding a substantial set of independent edges. Invoke an algorithm from [4] to obtain set  $\mathcal{P}$  of independent (in  $G$ )  $(M, 3)$ -paths. The set  $\mathcal{P}$  has the following property: If the number of  $(M, 3)$ -paths in  $G'$  which are incident to edges of  $\bigcup_{P \in \bar{\mathcal{P}}} E(P)$  is at least  $\alpha m'_3$  then at least  $\alpha m'_3/4$ ,  $(M, 3)$ -paths in  $G'$  are incident to edges of  $\bigcup_{P \in \mathcal{P}} E(P)$ .

---

Note that the resulting set  $\mathcal{P}$  is a set of  $(M, 3)$ -paths which are independent in  $G$ . In addition the set is path-substantial with a constant specified in the next lemma.

**Lemma 6.** *If a set of paths  $\bar{\mathcal{P}}$  is  $\xi$ -path-substantial in  $\bar{G}$  then the set of paths  $\mathcal{P}$  obtained by procedure TRANSLATE is  $\xi/16$ -path-substantial in  $G$ .*

*Proof.* The proof follows from Lemma 5 and Step (2) of TRANSLATE.

### 3.3 Main Procedure

Now we can summarize the discussion in our main algorithm. We denote by  $A \div B$  the symmetric difference between  $A$  and  $B$ .

---

### Procedure Matching

---

Input: Graph  $G$ .

Output: Matching  $M^*$  such that  $|M^*| \geq \frac{2}{3}\beta(G)$ .

1. Find a maximal matching  $M$  using the procedure MATCH from [4].
  2. Let  $\mathcal{P}_I := \emptyset$  and  $M^* := M$ .
  3. Iterate  $O(\log n)$  times:
    - (a) Invoke REDUCE to obtain a 4L-graph  $\bar{G}$ .
    - (b) Invoke PATHSIN4LGRAPH to obtain  $\bar{\mathcal{P}}$ .
    - (c) Use TRANSLATE to obtain  $\mathcal{P}$ .
    - (d) Use MODIFY to modify  $\bar{G}$  with respect to  $\mathcal{P}$ .
    - (e)  $\mathcal{P}_I := \mathcal{P}_I \cup \mathcal{P}$ .
  4.  $M^* := M^* \div (\bigcup_{P \in \mathcal{P}_I} E(P))$ .
-

*Proof (of Theorem 3).* First we establish the time complexity. We use  $O(\log^4 n)$  rounds in the first step, to find  $M$  using the algorithm from [4]. In each iteration, the main time complexity comes from PATHSIN4LGRAPH which, by Lemma 4 is  $O(\log^3 V(\bar{G})) = O(\log^3 n)$ . Since we have  $O(\log n)$  iterations, the number of steps is  $O(\log^4 n)$ .

By Lemma 4, the set of paths  $\bar{\mathcal{P}}$  found in each iteration is  $\frac{1}{256K}$ -path-substantial in  $\bar{G}$ . Therefore, by Lemma 6,  $\mathcal{P}$  is  $\frac{1}{4096K}$ -path-substantial in  $G$  and so in step (3d) a constant fraction of  $(M, 3)$ -paths will be deleted from  $G$ . Since the number of  $(M, 3)$ -paths in  $G$  is  $O(n^4)$ , there will be no  $(M, 3)$ -paths in  $G$  after  $O(\log n)$  iterations. Consequently, after step (3),  $\mathcal{P}_I$  is a maximal set of  $(M, 3)$ -paths. Thus, after exchanging edges in step (4), we obtain matching  $M^*$  such that there is no  $(M^*, 3)$ -path in  $G$ . Therefore, by Claim 4,  $|M^*| \geq \frac{2}{3}\beta(G)$ .

*Final Remarks.* An efficient randomized algorithm for approximating the maximum matching can be obtained by invoking a randomized MIS procedure ([7]) in an auxiliary graph where the vertex set consists of augmenting paths and two vertices are connected if the corresponding paths share a vertex in the original graph.

The augmenting paths technique can only be applied to an unweighted version of the problem; to approximate a maximum weighted matching a completely different approach must be taken.

## References

1. Awerbuch, B., Goldberg, A.V., Luby, M., Plotkin, S.: Network decomposition and locality in distributed computing. Proc. of the 30th Symposium on Foundations of Computer Science (FOCS 1989) 364–369
2. Czygrinow, A., Hańćkowiak, M., Szymańska, E.: Distributed algorithm for approximating the maximum matching. Discrete Applied Math. published online (March 19, 2004)
3. Fischer, T., Goldberg, A.V., Haglin, D. J., Plotkin, S.: Approximating matchings in parallel. Information Processing Letters **46** (1993) 115–118
4. Hańćkowiak, M., Karoński, M., Panconesi, A.: On the distributed complexity of computing maximal matchings. SIAM J. Discrete Math. Vol.15, No.1, (2001) 41–57
5. Lovász, L., Plummer, M.: Matching Theory. Elsevier, Amsterdam and New York (1986)
6. Linial, N.: Locality in distributed graph algorithms. SIAM Journal on Computing **21**(1) (1992) 193–201
7. Luby, M.: A Simple Parallel Algorithm for the Maximal Independent Set Problem. SIAM J. on Computing **15**(4) (1986) 254–278.