# Bipartite Matching is in NC

Bogdan Grechuk • 28 Jun 2026

In a recent preprint (Chatterjee et al. 2026), Chatterjee, Ghosh, Gurjar, Raj, and Thierauf developed an NC algorithm for finding a maximum-cardinality matching in a bipartite graph. This resolves one of the central open questions in the theory of deterministic parallel computation.

Modern computers contain many processors, so it is natural to ask which algorithmic tasks can be efficiently parallelized. Informally, a problem is efficiently parallelizable if a large number of processors can work simultaneously and finish the computation much faster than any known sequential algorithm. Formally, a decision problem belongs to the class NC if it can be solved by a deterministic parallel algorithm in polylogarithmic time using polynomially many processors. It follows directly from the definition that $\mathsf{NC} \subseteq \mathsf{P}$.

Strictly speaking, NC is a class of decision problems. For search problems, however, we will use the standard abuse of terminology and say that a search problem is in NC if its output can be produced within the same parallel resource bounds. The class NC contains many basic arithmetic problems, such as integer addition, multiplication, and division. For many other problems in P, however, the existence of a deterministic polylogarithmic-time parallel algorithm is far from obvious.

A particularly important example is the problem of finding a maximum-cardinality matching in a graph $G$. Recall that a matching is a set of edges no two of which share a common vertex. The maximum-cardinality matching problem asks, given $G$, to return a matching with as many edges as possible. If $G$ has $n$ vertices, then every matching has size at most $n/2$. A matching of size exactly $n/2$ is called a *perfect matching*; of course, such a matching can exist only when $n$ is even.

The first polynomial-time algorithm for finding a maximum-cardinality matching in a general graph was developed by Edmonds (Edmonds 1965) in 1965. In 1980, Micali and Vazirani (Micali and Vazirani 1980; Vazirani 2020) gave an algorithm running in time $O(m\sqrt{n})$ on $n$-vertex graphs with $m$ edges. In 2004, Mucha and Sankowski (Mucha and Sankowski 2004) developed a randomized algorithm running in time $O(n^\omega)$, where $\omega$ is the exponent of matrix

multiplication, currently known to be less than $2.38$. This algebraic algorithm is faster than the $O(m\sqrt{n})$ algorithm on sufficiently dense graphs, namely when $m > n^{\omega-1/2}$.

The randomized parallel history of the problem begins with the work of Lovász (Lovász 1979), who gave a randomized parallel algorithm for the decision version: given a graph $G$ and a positive integer $k$, decide whether $G$ has a matching of size at least $k$. The class of problems admitting randomized polylogarithmic-time parallel algorithms with polynomially many processors is called RNC. Lovász's algorithm, however, does not output a matching. In 1986, Karp, Upfal, and Wigderson (Karp et al. 1986) proved that the corresponding search problem also belongs to RNC: one can construct a maximum-cardinality matching by a randomized parallel algorithm. Later, Mulmuley, U. Vazirani, and V. Vazirani (Mulmuley et al. 1987) gave a cleaner and faster RNC algorithm for the same problem.

Standard derandomization conjectures would imply $\mathsf{NC} = \mathsf{RNC}$, but this equality remains open. Consequently, maximum-cardinality matching in general graphs has long been one of the most prominent problems known to be in RNC but not known to be in NC. For general graphs, the best deterministic parallel result currently known is due to Svensson and Tarnawski (Svensson and Tarnawski 2017): the problem can be solved by a deterministic parallel algorithm running in $O(\log^3 n)$ time using $n^{O(\log^2 n)}$ processors. Thus the running time is polylogarithmic, but the number of processors is quasi-polynomial rather than polynomial.

The difficulty of the general problem motivates the study of special graph classes, such as planar graphs and bipartite graphs. For planar graphs, much of the work focused first on the apparently simpler task of finding a *perfect* matching, if one exists. In 1989, Vazirani (Vazirani 1989) proved that the seemingly harder problem of *counting* perfect matchings in planar graphs is in NC. In 1995, Miller and Naor (Miller and Naor 1995) developed an NC algorithm for finding a perfect matching in bipartite planar graphs. Mahajan and Varadarajan (Mahajan and Varadarajan 2000) later gave a conceptually different NC algorithm for the same problem, using the NC algorithm for counting perfect matchings as a subroutine.

In 2020, Anari and Vazirani (Anari and Vazirani 2020) proved the analogous search result for general planar graphs.

**Theorem 1** *There is an* NC *algorithm which, given a planar graph $G$, either returns a perfect matching in $G$ or correctly reports that no perfect matching exists.*

The proof of Theorem 1 uses the NC counting subroutine for planar perfect matchings and extends ideas of Mahajan and Varadarajan (Mahajan and Varadarajan 2000). The non-bipartite case requires additional structural ingredients, in particular the use of tight odd sets in the planar perfect matching polytope.

However, Theorem 1 concerns only *perfect* matchings. It does not give an NC algorithm for finding a maximum-cardinality matching when no perfect matching exists. In fact, for non-bipartite planar graphs, it remains open whether maximum-cardinality matching is in deterministic NC.

For bipartite graphs, the situation has now changed. In a 2026 preprint, Chatterjee, Ghosh, Gurjar, Raj, and Thierauf (Chatterjee et al. 2026) resolved the deterministic parallel complexity of bipartite matching.

**Theorem 2** *There is an* NC *algorithm which, given a bipartite graph, returns a maximum-cardinality matching. More generally, for bipartite graphs with polynomially bounded integer edge weights, there is an* NC *algorithm which returns:*

1. *a maximum-weight matching; and*
2. *a maximum-weight perfect matching, if one exists, and otherwise correctly reports that no perfect matching exists.*

Theorem 2 is a major advance because it removes randomness from one of the classical parallel algorithms for matching. Before this work, bipartite matching was known to have efficient randomized parallel algorithms, but no deterministic NC algorithm was known for the full maximum-cardinality problem. The new result shows that, at least in the bipartite case, the randomness used in earlier parallel matching algorithms is not essential.

This also clarifies the landscape around matching problems. Perfect matching in planar graphs is in NC, maximum-cardinality matching in bipartite graphs is now in NC, and maximum-cardinality matching in general graphs remains one of the central unresolved problems in deterministic parallel computation. The remaining gap is therefore not merely technical: it marks the boundary between what is currently understood and what is still mysterious about the parallel structure of matching.

# References

Anari, Nima, and Vijay V. Vazirani. 2020. “Planar Graph Perfect Matching Is in NC.” *J. ACM* 67 (4): 1–34.

Chatterjee, Abhranil, Sumanta Ghosh, Rohit Gurjar, Roshan Raj, and Thomas Thierauf. 2026. *Bipartite Matching Is in NC*. No. 100. Electronic

Colloquium on Computational Complexity (ECCC). https://eccc.weizmann.ac.il/report/2026/100/.

Edmonds, Jack. 1965. “Paths, Trees, and Flowers.” *Canadian J. Math.* 17: 449–67. https://doi.org/10.4153/CJM-1965-045-4.

Karp, Richard M., Eli Upfal, and Avi Wigderson. 1986. “Constructing a Perfect Matching Is in Random NC.” *Combinatorica* 6 (1): 35–48.

Lovász, L. 1979. “On Determinants, Matchings, and Random Algorithms.” *Fundamentals of Computation Theory (Proc. Conf. Algebraic, Arith. And Categorical Methods in Comput. Theory, Berlin/Wendisch-Rietz, 1979)*, Math. res., vol. 2: 565–74.

Mahajan, Meena, and Kasturi R. Varadarajan. 2000. “A New NC-Algorithm for Finding a Perfect Matching in Bipartite Planar and Small Genus Graphs.” *Proceedings of the thirty-second annual ACM symposium on the Theory of Computing*, 351–57.

Micali, Silvio, and Vijay V Vazirani. 1980. “An $O(\sqrt{|v|}|E|)$ Algoithm for Finding Maximum Matching in General Graphs.” *21st Annual Symposium on Foundations of Computer Science (Sfcs 1980)*, 17–27.

Miller, Gary L., and Joseph Naor. 1995. “Flow in Planar Graphs with Multiple Sources and Sinks.” *SIAM J. Comput.* 24 (5): 1002–17.

Mucha, Marcin, and Piotr Sankowski. 2004. “Maximum Matchings via Gaussian Elimination.” *45th Annual IEEE Symposium on Foundations of Computer Science*, 248–55.

Mulmuley, Ketan, Umesh V. Vazirani, and Vijay V. Vazirani. 1987. “Matching Is as Easy as Matrix Inversion.” *Combinatorica* 7 (1): 105–13. https://doi.org/10.1007/BF02579206.

Svensson, Ola, and Jakub Tarnawski. 2017. “The Matching Problem in General Graphs Is in Quasi-NC.” In *58th Annual IEEE Symposium on Foundations of Computer Science—FOCS 2017*. IEEE Computer Soc., Los Alamitos, CA. https://doi.org/10.1109/FOCS.2017.70.

Vazirani, Vijay V. 2020. “A Proof of the MV Matching Algorithm.” *arXiv Preprint arXiv:2012.03582*.

Vazirani, Vijay V. 1989. “NC Algorithms for Computing the Number of Perfect Matchings in $K_{3,3}$-Free Graphs and Related Problems.” *Infor. and Comput.* 80 (2): 152–64.