$$\mathrm{PM}_n(x) = \bigvee_{\sigma \in S_m} \bigwedge_{i=1}^{m} x_{i,\sigma(i)},$$

where S_m is the set of all $m!$ permutations of $1, 2, \ldots, m$.

The *exact perfect matching* function EPM_n accepts a graph E iff E *is* a perfect matching. That is, EPM_n takes a boolean $n \times n$ matrix as an input, and outputs 1 iff this is a permutation matrix, that is, each row and each column has exactly one 1.

In Sect. 9.11 we have shown that PM_n requires monotone circuits of super-polynomial size. Now we show that it also requires 1-NBP of exponential size.

Corollary 16.11. *Every* 1-NBP *computing* PM_n *as well as any null-path-free* NBP *computing* EPM_n *must have size* $2^{\Omega(n)}$.

Proof. Let A be the set of all $|A| = n!$ permutation matrices; hence, A is m-uniform with $m = n$. Since only $(n-k)!$ perfect matchings can share k edges in common, we have that $d_k(A) = (n-k)!$. In particular, taking $k = n/2$, we obtain that $d(A) \le (n/2)! \cdot (n/2)!$. Observe that every program computing PM_n majorities A, and every program computing EPM_n must isolate A. Thus, Theorem 16.9 yields the desired lower bound $n!/d(A) \ge \binom{n}{n/2}$. □

To better understand the role of null-paths, we have to first solve the following problem. Say that a nondeterministic branching program is *weakly read-once* if along any *consistent* s-t path no variable is tested more than once. That is, we now put no restrictions on inconsistent paths: only consistent paths are required to be read-once.

The following problem is one of the "easiest" questions about branching programs, but it still remains open!

■ **Research Problem 16.12.** Prove an exponential lower bound for weakly read-once nondeterministic branching programs.

That such programs may be much more powerful than 1-NBPs shows the following observation made in Jukna (1995).

Proposition 16.13. *The function* EPM_n *can be computed by a weakly read-once nondeterministic branching program of size* $\mathcal{O}(n^3)$.

Proof. To test that a given square 0-1 matrix is a permutation matrix, it is enough to test whether every row has at least one 1, and every column has at least $n-1$ zeros. These two tests can be made by two nondeterministic branching programs P_1 and P_2 designed using the formulas

$$P_1(X) = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n} x_{i,j} \quad \text{and} \quad P_2(X) = \bigwedge_{j=1}^{n} \bigvee_{k=1}^{n} \bigwedge_{\substack{i=1 \\ i \ne k}}^{n} \neg x_{i,j}.$$