An improved fully polynomial randomized approximation scheme (FPRAS) for counting the number of Hamiltonian cycles in dense digraphs

Jinshan Zhang *, Fengshan Bai
Department of Mathematical Sciences, Tsinghua University, Beijing 100084, China

**Abstract**

We propose an improved algorithm for counting the number of Hamiltonian cycles in a directed graph. The basic idea of the method is sequential acceptance/rejection, which is successfully used in approximating the number of perfect matchings in dense bipartite graphs. As a consequence, a new ratio of the number of Hamiltonian cycles to the number of 1-factors is proposed. Based on this ratio, we prove that our algorithm runs in expected time of $O(n^{8.5})$ for dense problems. This improves the Markov chain Monte Carlo method, the most powerful existing method, by a factor of at least $n^{4.5} (\log n)^4$ in running time. This class of dense problems is shown to be nontrivial in counting, in the sense that they are #P-Complete.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

A Hamiltonian cycle is a closed directed path that visits each vertex once and only once. In this paper, we use the term digraph to denote a directed graph. Counting the number of Hamiltonian cycles is a very challenging problem, and it has applications, for example, in quantum physics [4]. Many intractable counting problems have been added to Valiant’s [20] list of #P-Complete, which naturally corresponds to the concept NP-Complete for decision problems. Efficient approximating schemes called fully polynomial randomized approximation schemes (FPRASs) are naturally considered for these hard counting problems. If $M$ is the true value, a randomized algorithm is called an FPRAS if it takes time that is polynomial in the size of inputs, $\epsilon^{-1}$ and $\log \delta^{-1}$, to obtain an output $\tilde{M}$. Here $\tilde{M}$ is the approximation of $M$, satisfying

$$P((1-\epsilon)M \leq \tilde{M} \leq (1+\epsilon)M) \geq 1-\delta.$$ 

Due to the fact that the decision problem of whether a graph contains a Hamiltonian cycle is NP-Complete, there would be no FPRAS for counting the Hamiltonian cycles for general graphs unless NP = RP. Thus FPRASs for counting Hamiltonian cycles are only possible for special or restricted graphs, for example, elementary recursive algorithms [17] for random digraphs; Markov chain Monte Carlo (MCMC) methods for dense undirected graphs [6], for some random digraphs [8] and random regular graphs [5,7].

The sequential acceptance/rejection method was introduced by Huber [9] for counting the number of the perfect matchings in a dense regular bipartite graph. Recently, the regularity requirement was removed [10]. The primary tool used in the algorithm is the generalized Bregman bound [3] and the matrix scaling method.
The MCMC algorithm presented for random digraphs in [8] can be naturally extended to dense digraphs. This algorithm is based on sampling 1-factors of the digraphs and uses the self-reducing method [13] to approximate the counting. Recently, Bezáková et al. presented an algorithm that approximates the number of 1-factors in \(O(n^{4+1/(2\alpha-3/2)})\) expected time, via an accelerating simulated annealing technique [1].

In this paper, the ratio of the number of 1-factors to the number of Hamiltonian cycles is established to be \(O(n^{1+1/(2\alpha-3/2)})\), provided that the digraph is \(\alpha n\) dense. Due to this ratio and Bezáková’s results in [1], the MCMC method [8] runs in an \(O(n^{6.5+1/(2\alpha-1)}+1/(2\alpha-1.5))\) \(\epsilon^{-2} \log(\delta^{-1})\). In particular, when \(\alpha \geq 0.85\), the running time is bounded by \(O(n^{8.5})\).

Our algorithm for counting Hamiltonian cycles is based on the acceptance/rejection algorithm in [10], while a different sequential sampling procedure is constructed to ensure that the approximating target is the number of Hamiltonian cycles.

One of the remarkable advantages of the acceptance/rejection method is that it samples perfectly from a given set, which removes the sampling error when the MCMC method is adopted. Hence, our algorithm generates a weighted Hamiltonian cycle exactly according to its weight from the set of Hamiltonian cycles of a weighted digraph. In addition, this perfect sampling is only by-product when acceptance/rejection is used to approximate counting, which means that the time used to sample a random Hamiltonian cycle can be used to approximate the number of Hamiltonian cycles without extra cost.

The main result of this paper is summarized in the following.

**Theorem M.** For any \(\epsilon, \delta \in (0, 1)\) and \(\alpha \in (0.75, 1]\), there exists a randomized approximation algorithm which provides an FPRAS for computing the number of Hamiltonian cycles of \(\alpha n\) dense digraphs. The same algorithm approximates the number of Hamiltonian cycles by a factor in \([1-\epsilon, 1+\epsilon]\) with probability at least \(1-\delta\) and has the complexity \(O(n^{2.5+1/(2\alpha-1)}+1/(2\alpha-1.5))\) \(\epsilon^{-2} \log(\delta^{-1})\). In particular, when \(\alpha \geq 0.85\), the running time is bounded by \(O(n^{8.5})\).

The remainder of the paper is structured as follows. In Section 2, some basic definitions, notations and lemmas are presented. In Section 3, we describe the algorithm in detail. Section 4 discusses the complexity of the algorithm and the hardness of counting. Further discussion and our conclusions are proposed in Section 5.

## 2. Preliminaries

Consider a simple weighted digraph \(G = (V, E)\) with the vertex set \(V = \{1, \ldots, n\}\) and the edge set \(E\). Each edge \((i, j) \in E\) is endowed with a positive weight \(w_{ij}\). Let \(|S|\) denote the cardinality of any set \(S\). The set of vertices pointing to \(i\) is denoted by \(N^{-}(i, G) = \{j : (i, j) \in E\}\), and similarly that out of \(i\) by \(N^{+}(i, G) = \{j : (i, j) \in E\}\). The indegrees and outdegrees of the vertex \(i\) are denoted by \(\Delta^{-}(i) = |N^{-}(i, G)|\) and \(\Delta^{+}(i) = |N^{+}(i, G)|\), respectively. Let \(\Delta(i) = \min(\Delta^{-}(i), \Delta^{+}(i))\) and \(\Delta = \min_{i \in V} \Delta(i)\). \(G\) is called \(\alpha n\) dense if \(\Delta \geq \alpha n\) for an \(\alpha > 0\) given. Let \(\oplus\) denote the symmetric difference of two sets and \([n]\) denote the maximum integer no more than \(n\). \(A/B\) is used to denote the set obtained by removing elements of \(B\) from set \(A\). With a little abuse of notation, \(\sigma\) also denotes the quotient of two numbers. A Hamiltonian cycle in \(G\) is represented by

\[H = (k_1, k_2, \ldots, k_n, k_1),\]

where \((k_1, k_2, \ldots, k_n)\) is a permutation of \(\{1, \ldots, n\}\) such that \((k_a, k_1) \in E\) and \((k_j, k_{j+1}) \in E, j = 1, \ldots, n-1\). The length of a cycle or path is defined as the number of its edges that contains.

A 1-factor is defined as a spanning directed subgraph of \(G\) in which the indegrees and outdegrees of each vertex are all 1. An example of a 1-factor is a spanning union of vertex disjoint directed cycles. Obviously, a Hamiltonian cycle is a special 1-factor with only one cycle. The weight \(W(F)\) of a 1-factor \(F\) with edge set \(\{e \in E\}_{\text{eff}}\) is defined as \(W(F) = \prod_{e \in F} w_e\). The total weight \(W(S)\) of the set \(S\) of 1-factors is defined as \(W(S) = \sum_{F \in S} W(F)\). Let \(W_{\mathcal{F}}(G)\) and \(W_{\mathcal{H}}(G)\) denote the total weight of all the 1-factors and Hamiltonian cycles in \(G\), respectively. It is easy to see that, if \(w_{ij} = 1\) for all \((i, j) \in E\), then \(W_{\mathcal{F}}(G)\) and \(W_{\mathcal{H}}(G)\) are the numbers of 1-factors and Hamiltonian cycles in \(G\), respectively.

Let \(A_{G}\) be the adjacent matrix associated with \(G\), where \(A_{G}(i, j) = w_{ij}\) if \((i, j) \in E\) and \(A_{G}(i, j) = 0\) otherwise. For an \(n \times n\) matrix \(A\), where \(n\) is the order of \(A\), we use notation \(A_{ij}\) to denote the \((n-1) \times (n-1)\) matrix obtained from \(A\) by removing row \(i\) and column \(j\). If there is no confusion, \(A_{ij}\) or \((A)_{ij}\) denotes the \((n-1) \times (n-1)\) matrix obtained from \(A\) after first permuting row \(i\) and row \(j\) and then removing row \(j\) and column \(j\). For example, the following is how \(A\) becomes \(A_{31}\).

\[
A = \begin{bmatrix}
1 & 2 & 3 \\
0 & 4 & 5 \\
0 & 0 & 6
\end{bmatrix} \Rightarrow \begin{bmatrix}
0 & 0 & 6 \\
0 & 4 & 5 \\
1 & 2 & 3
\end{bmatrix} \Rightarrow A_{31} = \begin{bmatrix}
4 & 5 \\
2 & 3
\end{bmatrix}.
\]

Next we will define two quantities on the matrix \(A_{G}\) which are related to 1-factors and Hamiltonian cycles, respectively.

**Definition 1.** The permanent of an \(n \times n\) matrix \(A = (A(i, j))_{n \times n}\) is

\[
\text{per}(A) = \sum_{\sigma} \prod_{i=1}^{n} A(i, \sigma(i)),
\]

where \(\sigma\) ranges over all the permutations of \(\{1, \ldots, n\}\).
The Hamilton of an $n \times n$ matrix $A = (A(i, j))_{n \times n}$ is defined as

$$\text{ham}(A) = \sum_{|k|} A(k_1, 1)A(k_2, k_1) \cdots A(k_{n-1}, k_{n-2})A(1, k_{n-1}),$$

where $\{k_1, \ldots, k_{n-1}\}$ ranges over all the permutations of $\{2, \ldots, n\}$ when $n \geq 2$, and $\text{ham}(A) = A(1, 1)$, if $n = 1$.

Since Definition 2 is a key for the approximation algorithm, we present an example of it. Take

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

for example. It is easy to see that

$$\text{ham}(A) = A(2, 1) * A(3, 2) * A(1, 3) + A(3, 1) * A(2, 3) * A(1, 2)$$

$$= 4 * 8 * 3 + 7 * 6 * 2 = 180.$$

By the definition of the permanent and the Hamilton, it is not difficult to see that $\text{per}(A) \geq \text{ham}(A)$ if the entries of $A$ are all nonnegative. Suppose that $A = A_C$. For any permutation $(k_1, \ldots, k_{n-1})$ of $(2, \ldots, n)$, $A_C(1, k_{n-1}), A_C(k_{n-1}, k_{n-2}), \ldots, A_C(k_2, k_1), A_C(k_1, 1)$ are the edge weights of the Hamiltonian cycle $(1, k_{n-1}, \ldots, k_1, 1)$ in $G$ if and only if they are all positive. Therefore, we have

$$W_H(G) = \text{ham}(A_C).$$

Note that the diagonal entries of $A_C$ are all zero, and for any permutation $\sigma$ over $\{1, 2, \ldots, n\}$, $A(i, \sigma(i)) > 0$, $i = 1, \ldots, n$ if and only if their corresponding edges in $G$ form a 1-factor of $G$. Hence

$$W_F(G) = \text{per}(A_C).$$

Next we present the expansion formulas for the permanent and the Hamilton.

**Lemma 3.** Let $A = (A(i, j))_{n \times n}$ be an $n \times n$ matrix. The permanent of empty matrix is set to be 1. Then

$$\text{per}(A) = \sum_{i=1}^{n} A(i, 1) \text{per}(A_{i1}).$$

**Lemma 4.** Let $A = (A(i, j))_{n \times n}$ be an $n \times n$ matrix, $n \geq 2$. Then

$$\text{ham}(A) = \sum_{i=2}^{n} A(i, 1) \text{ham}(A_{i1}).$$

For the permanent, this expansion is well known. For the Hamilton, the formula is very similar, and [17] proposes a combinatorial proof when each edge weight of the digraph is 1. Because of its importance to our algorithm, a proof in matrix terms is presented below. We emphasize that Lemma 4 is crucial in the sequential sampling procedure, which is different from the one used in [10], and ensures that our algorithm approximates the number of Hamiltonian cycles.

**Proof of Lemma 4.** We proceed to prove the lemma by induction on $n$, the order of the matrix.

The case $k = 2$ is trivial.

Suppose that Lemma 4 holds for $k = n - 1$.

Consider $k = n$. Since

$$\text{ham}(A) = \sum_{i=2}^{n} A(i, 1) \sum_{|k|} A(k_2, i)A(k_3, k_2) \cdots A(k_{n-1}, k_{n-2})A(1, k_{n-1}),$$

it is sufficient to show that

$$\text{ham}(A_{i1}) = \sum_{|k|} A(k_2, i) \cdots A(k_{n-1}, k_{n-2})A(1, k_{n-1}),$$

for $i = 2, \ldots, n$, where $\{k_2, \ldots, k_{n-1}\}$ goes over all the permutations of $(2, \ldots, n)/i$. Considering the definition of $A_{i1}$, the row $i - 1$ of $A_{i1}$ is the first row of $A$ except removing the first element, and

$$A_{i1}(k_2 - 1, i - 1) = A(k_2, i), \ldots,$$

$$A_{i1}(k_{n-1} - 1, k_{n-2} - 1) = A(k_{n-1}, k_{n-2}), \quad \text{and}$$

$$A_{i1}(i - 1, k_{n-1} - 1) = A(1, k_{n-1}).$$
By the hypothesis of the induction, the order of $A'_{1}$ is $n - 1$; then
\[
\text{ham}(A'_{i}) = \sum_{k_{2}, \ldots, k_{n-1}} A'_{i}(k'_{2}, i - 1) \cdots A'_{i}(k'_{n-1}, k'_{n-2}) A'_{i}(i - 1, k'_{n-1})
\]
\[
= \sum_{k_{2}, \ldots, k_{n-1}/i} A'_{i}(k_{2} - 1, i - 1) A'_{i}(k_{n-1} - 1, k_{n-2} - 1) A'_{i}(i - 1, k_{n-1} - 1)
\]
\[
= \sum_{k_{2}, \ldots, k_{n-1}/i} A(k_{2}, i) \cdots A(k_{n-1}, k_{n-2}) A(k_{n-1}, k_{n-1}),
\]
where $\{k'_{2}, \ldots, k'_{n-1}\}$ and $\{k_{2}, \ldots, k_{n-1}\}$ go over all the permutations of $\{1, \ldots, n - 1\}/\{i - 1\}$ and $\{2, \ldots, n\}/\{i\}$, respectively. This completes the proof of Lemma 4. □

**Hamiltonian recovery.** Let $A = (A(i, j))_{1 \times n}$ be an $n \times n$ positive matrix. The following procedure is applied to select elements from $A$ (the first two steps are given explicitly). We call this procedure selecting the Hamiltonian cycle (SHC for simplicity).

**Step 1.** Let $A_{1} = A$. Choose a natural number $1 < j_{1} \leq n$, denote $\pi(1) = j_{1}$, and select $A^{1}(\pi(1), 1)$.

**Step 2.** Let $A^{2} = (A^{1})'_{j_{1}1}$ (recall the definition of $(A^{1})'_{j_{1}1}$, which means first permutating row $j_{1}$ and row 1 of $A^{1}$ and then removing row 1 and column 1 of $A^{1}$; here $(A^{1})'_{11} = (A)_{11} = A_{11}$). Choose a natural number $1 < j_{2} \leq n - 1$, denote $\pi(2) = j_{2}$, and select $A^{2}(\pi(2), 1)$.

Iteratively in Step $k$, $A^{k} = (A^{k-1})'_{j_{k}1}$. Similarly, $\pi(k)$ and $A^{k}(\pi(k), 1)$, $1 \leq k \leq n - 1$, can be obtained in this step. Since $A^{n}$ has only one entry, let $A^{n} = (A^{n-1})'_{j_{n-1}1}$, $\pi(n) = j_{n} = 1$, and select $A^{n}(\pi(n), 1)$.

By Lemma 4, the set of selected elements $A^{k}(\pi(k), 1), k = 1, \ldots, n$, from the above procedure forms the edge weight of a Hamiltonian cycle in $G$ if $A = A_{G}$. If $\pi(1), \pi(2), \ldots, \pi(n)$ is given, we provide a simple algorithm to determine which Hamiltonian cycle in $G$ is selected. This process is called Hamiltonian Recovery.

The input of the algorithm is $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$. We illustrate how to recover an entry in $A^{2}$ if $\pi(1)$ is given. Let $A^{2}(i, j)$ be any entry in $A^{2}$. $A^{2} = (A^{1})'_{\pi(1)},$ and recall the definition of $(A^{1})'_{\pi(1)}$, which is obtained by first permutating row $\pi(1)$ and the first row and then removing the first row and first column. Hence, if $i = \pi(1) - 1$, then $(1, j + 1)$ is the position where $A^{2}(i, j)$ lies of $A^{1}$; otherwise $(i + 1, j + 1)$ is the position where $A^{2}(i, j)$ lies in $A^{1}$. Hence, if the vector $(\pi(1), \ldots, \pi(k-1))$ is given from the SHC procedure, the position of $A^{k}(\pi(k), 1)$ in $A^{1}$ can be found recursively by determining its position in $A^{k-1}$, then in $A^{k-2}$, and finally in $A^{1}$. Since at each step of the SHC procedure an element is selected from the first column, $A^{k}(\pi(k), 1)$ must lie in column $k$ of $A^{1}$.

If $(1, k_{1}, \ldots, k_{n-1}, 1)$ is the corresponding Hamiltonian cycle of $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$, then $k_{1}$ can be obtained from $k_{i+1}$ since the element $A(k_{i}, k_{i+1})$ is selected in Step $k_{i+1}$ of the SHC procedure, or equivalently $(k_{i}, k_{i+1})$ is the position of $A^{k_{i+1}}(\pi(k_{i+1}), 1)$ in $A$, $i = 1, 2, \ldots, n - 2$. Obviously, $k_{n-1} = \pi(1)$. By this simple procedure, it takes $O(n^2)$ time to recover all the positions of $A^{k}(\pi(k), 1), 1 \leq k \leq n$. We present the recovery algorithm explicitly.

**Hamiltonian Recovery Algorithm**

**Input:** The vector $(\pi(1), \pi(2), \ldots, \pi(n))$.

**Output:** A Hamiltonian cycle $(1, k_{1}, \ldots, k_{n-1}, 1)$.

**Step 1:** Set $k_{n-1} = \pi(1)$.

**For** $i = n - 2$ to 1

- Set $a = \pi(k_{i+1})$;
- For $j = k_{i+1}$ to 2
  - If $a = \pi(j - 1) - 1$; Set $a = 1$;
  - Else Set $a = a + 1$;
- End;
- Set $k_{i} = a$;
**End;**

- Goto Step 2;
**Step 2:** Output $(1, k_{1}, \ldots, k_{n-1}, 1)$.

Take a matrix of order $n = 3$, for example. If the input vector of the Hamiltonian Recovery Algorithm is the vector $(3, 2, 1)$, then the output of Step 2 of the above algorithm is the Hamiltonian cycle $(1, 2, 3, 1)$.

For simplicity, let $HR(\pi)$ denote the output of the Hamiltonian Recovery Algorithm when the input is $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$.

3. Algorithms for counting

One main tool in our algorithm is a generalized version of Bregman's bound for the permanent below, which generalized an inequality of Soul [19] and was proved in [10]. For more applications of other generalizations of Bregman’s bound for designing new algorithms or improving the efficiency of algorithms, we refer to [14,18]. Let

$$h(r) = \begin{cases} r + (1/2) \log r + e - 1, & r \geq 1 \\ 1 + (e - 1)r, & r \in [0, 1]. \end{cases}$$
Lemma 5 ([10]). Let $A$ be an $n \times n$ matrix with entries in $[0, 1]$. Denote $r(i)$ the sum of row $i$ of $A$. Define $Br(A) = \prod_{i=1}^{n}(h(r(i))/e)$; then

$$Br(A) \geq \sum_{i=1}^{n} A(i, 1)Br(A_{ii}).$$

In particular, by Lemma 3, $per(A) \leq Br(A)$.

Chernoff’s bound is useful in our algorithm, and one form of it is given below [16].

Lemma 6. Let $x_1, x_2, \ldots, x_n$ be identical independently distributed (i.i.d.) Bernoulli random variables with $P(x_1 = 1) = p$ and $P(x_1 = 0) = 1 - p$, $p > 0$; then, for any $0 \leq \varepsilon \leq 2e - 1$,

$$P \left( \frac{\sum_{i=1}^{n} x_i - np}{\varepsilon np} > \frac{e np}{2} \right) \leq e^{-n\varepsilon^2/4}.$$

For simplicity, in this section, we only consider the digraph $G$ with all edge weights equal to 1. Hence the adjacent matrix $A_C$ is a 0–1 matrix and $\text{ham}(A_C)$ is the number of Hamiltonian cycles in $G$. $G$ is also restricted to be an $n$ dense, $\alpha \geq 0.75$. It is known [2] that if $G$ is .5n dense, $G$ must contain a Hamiltonian cycle, and the proof can be easily modified to give an $O(n^2)$ algorithm to construct a Hamiltonian cycle. Hence $\text{ham}(A_C) \geq 1$. By the definition of the Hamilton, if we change any zeros in $A_C$ to $\gamma = (\varepsilon/3)((n - 1)!)^{-1}$, $\text{ham}(A_C)$ increases by at most a factor of $1 + \varepsilon/3$.

Now we introduce the basic idea of the acceptance/rejection method for the counting problem. Suppose that $S$ is a large set and that each element in it has positive weight. The target is to approximate the total weight of all the elements in $S$. First, select a suitably large $M$ such that $M > \sum_{b \in S} w(b)$, the main idea of the acceptance/rejection method for approximation is to design a procedure to sample a random element $x$ from the set $S$ with successful probability $P(x = a) = \frac{w(a)}{M}$, where $w(a)$ is the weight of $a \in S$, and failing probability $P(x \notin S) = 1 - \sum_{b \in S} \frac{w(b)}{M}$. Each time, if a random element $a$ is successfully selected from $S$, we say that acceptance occurs, or it is accepted, and if no element is selected from $S$, we say that rejection occurs. Hence, each time, the probability of acceptance is $\sum_{b \in S} \frac{w(b)}{M}$ and the probability of rejection is $1 - \sum_{b \in S} \frac{w(b)}{M}$. With some fundamental statistical knowledge, the total weight of $S$ can be approximated by multiplying $M$ and the ratio of acceptance over all the samplings. For our purposes, the generalized Bregman bound in Lemma 5 provides such a suitable large $M$, and the self-reducing method for counting Hamiltonian cycles naturally proposes such a sampling procedure, which is a sequential sampling procedure guaranteed by Lemma 4. For more details about the sequential acceptance/rejection method, refer to [10].

In order to make use of the generalized Bregman bound in Lemma 5, before resuming the acceptance/rejection algorithm, we need to scale the matrix $A_C$ to nearly be doubly stochastic and make each entry in $[0, 1]$ [15]. Hence the algorithm has two phases.

Sub Algorithm I. Scale Matrix

Input: $A_C$, $\varepsilon$

Output: $X, Y, Z, C$

Step 1: Set $A_C(i, j) = (\varepsilon/3)((n - 1)!)^{-1}$ if $A_C(i, j) = 0$ for all $i, j$, goto Step 2;

Step 2: Using matrix scaling to find diagonal matrix $X, Y$ such that the row and column sums of $B = XA_CY$ in $(1 - (1/2)n^{-2}, 1 + (1/2)n^{-2})$, goto Step 3;

Step 3: Let $Z$ be a diagonal matrix with $Z(i, i) = \min_j B(i, j)^{-1}$ for $i = 1, \ldots, n$, goto Step 4;

Step 4: $C = ZB$.

After matrix scaling, matrix $C$ satisfies the requirement of the generalized Bregman bound. The sequential acceptance/rejection method can be used to estimate $\text{ham}(C)$. Note that the matrix $C$ corresponds to a weighted digraph denoted by $GC$.

Sub Algorithm II. Approximating Hamilton via Acceptance/Rejection

Input: $X, Y, Z, C, \varepsilon, \delta, N$

Output: $\hat{H}_1, \ldots, \hat{H}_s$, $\text{ham}(A_C)$ the estimator of $\text{ham}(A_C)$.

Step 5: Set $t = 4N(\varepsilon/2)^{-2} \log(\delta^{-1})$, $l = \prod_{i=1}^{n}(X(i, i)Y(i, i)Z(i, i))$, $D = C$, $k = 0$ and $s = 0$, goto Step 6;

Step 6: Set $r = \text{order of } D$;

If $r = 1$;

Set $p(1) = D/Br(D)$ and $p(0) = 1 - p(1)$;

Choose $l$ from $[0, 1]$ according to $P(l = i) = p(i)$, $i = 0, 1$;

If $l > 0$; Set $\pi(n) = 1$, $s = s + 1$, $k = k + 1$ and $\hat{H}_s = HR(\pi)$;

If $k < t$; Set $D = C$, goto Step 6; Otherwise goto Step 7;

Else $l = 0$; Set $k = k + 1$;

If $k < t$; Set $D = C$, goto Step 6; Otherwise goto Step 7;

Else $r > 1$;

Set $p(i) = D(i, 1)Br(D')/Br(D)$ for $i = 2, \ldots, r$ and $p(0) = 1 - \sum_{i=2}^{r} p(i)$;
Choose $l$ from $\{0, 2, 3, \ldots, r\}$ according to $P(l = i) = p(i), i = 0, 2, \ldots, r$;
If $l > 0$; Set $\pi(n + 1 - r) = l$ and $D = D'_1$, goto Step 6;
Else $l = 0$; Set $k = k + 1$
If $k < t$; Set $D = C$, goto Step 6; Otherwise goto Step 7;

**Step 7:** $\widehat{\text{ham}}(A_C) = t^{-1} \text{st}^{-1} \text{Br}(C)$.

The procedure of sampling elements in Step 6 is the same as the SHC procedure apart from selecting an element with certain probability or rejection when $l = 0$ is selected. The output $\mathcal{H}_i$, $1 \leq i \leq s$, is accepted by the algorithm.

**Theorem 7.** Let $\mathcal{H}_1, \ldots, \mathcal{H}_s$ and $\widehat{\text{ham}}(A_C)$ be the output of Sub Algorithm II. If we set $N = \text{Br}(C)/ \text{ham}(C)$ in the same algorithm, and let $H$ be a random variable recovered from a random $\pi$ of Sub Algorithms II and let $S$ denote the set of all the possible accepted Hamiltonian cycles, then

$$P(H = \mathcal{H}_1 | H \in S) = W(\mathcal{H}_1)/W_H(G_C)$$

and

$$P((1 - \varepsilon) \text{ ham}(A_C) \leq \widehat{\text{ham}}(A_C) \leq (1 + \varepsilon) \text{ ham}(A_C)) \geq 1 - \delta.$$  

**Proof.** First, we check that $p(0) \geq 0$ at each level of Step 6, which guarantees the proceeding of the algorithm. By the definition of $D'_i$ and $D_i$, obviously, $\text{Br}(D'_i) = \text{Br}(D_i)$. Using Lemma 5, it is easy to see that

$$\sum_{i=2}^{n} D(i, 1) \text{Br}(D'_i) = \sum_{i=2}^{n} D(i, 1) \text{Br}(D_i) \leq \sum_{i=1}^{n} D(i, 1) \text{Br}(D_i) \leq \text{Br}(D).$$

Hence $p(0) \geq 0$. Suppose that $\mathcal{H}_1 = HR(j)$, $j = (j_1, \ldots, j_n)$. Following the path in which $\mathcal{H}_1$ is selected, and using the notation in the SHC procedure, then $c^{i+1} = c^i j_i, i = 1, \ldots, n - 1, \text{ and } C^1 = C$, and we have

$$P(\pi(k) = j_k) = \frac{C^k(j_k, 1) \text{Br}((C^k)_j)}{\text{Br}(C^k)},$$

where $k = 1, 2, \ldots, n - 1$ and $P(\pi(n) = j_n) = \frac{c^n(j_n, 1)}{\text{Br}(C^n)}$.

Since the selection at each level in Step 6 is independent of the other, the probability of selecting $\mathcal{H}_1$ is the telescoping product. Noting that $c^{i+1} = (C^i)_j, i = 1, \ldots, n - 1, \text{ and } C^1 = C$, then

$$P(H = \mathcal{H}_1) = P(\pi = j) = \prod_{k=1}^{n} P(\pi(k) = j_k) = \frac{\prod_{i=1}^{n} C^i(j_i, 1)}{\text{Br}(C)} = \frac{W(\mathcal{H}_1)}{\text{Br}(C)}.$$ 

Since each Hamiltonian cycle in $G_C$ can be accepted with certain probability proportional to its weight, the acceptance set $S$ is the set of all the Hamiltonian cycles in $G_C$. Then

$$P(H \in S) = \sum_{\mathcal{H} \in G_C} P(H = \mathcal{H}) = \frac{W_H(G_C)}{\text{Br}(C)}.$$ 

Hence,

$$P(H = \mathcal{H}_1 | H \in S) = \frac{W(\mathcal{H}_1)}{W_H(G_C)}.$$ 

In Sub Algorithm II, let $x_k, 1 \leq k \leq t$, denote the indicator function of acceptance or rejection in Step 6; that is, $x_k = 1$ if a Hamiltonian cycle is accepted and $x_k = 0$ otherwise. Obviously, $x_k, 1 \leq k \leq t$, are i.i.d. Bernoulli random variables with $P(x_1 = 1) = p = W_H(G_C)/\text{Br}(C) = \text{ham}(C)/\text{Br}(C)$. Let $A'_C$ be the matrix obtained in Step 1 of Sub Algorithm I. Hence, by Lemma 6, and noting that $t = 4N(\varepsilon/2)^{-2} \log(\delta^{-1})$, where $N = \text{Br}(C)/\text{ham}(C)$, a simple calculation shows that

$$P((1 - \varepsilon/2) \text{ ham}(A'_C) \leq \widehat{\text{ham}}(A_C) \leq (1 + \varepsilon/2) \text{ ham}(A'_C)) \geq 1 - \delta.$$  

Noting that $\text{ham}(A_C) \leq \text{ham}(A'_C) \leq (1 + \varepsilon/3) \text{ ham}(A_C)$ completes the proof. \(\square\)

4. Complexity and hardness of counting

4.1. Complexity of the algorithm

By using the ellipsoid method [15], the running time of matrix scaling is $O(n^4 \log n)$. So the complexity of Sub Algorithm I is $O(n^4 \log n)$. 


The time of repeating Step 6 in Sub Algorithm II is \( t = O(\text{Br}(C)/\text{ham}(C)) \), and each time the running time is \( O(n^2) \); hence, the complexity of Sub Algorithm II is \( O(n^2 \cdot t) = O(n^3 \text{Br}(C)/\text{ham}(C)) \), where \( \epsilon^{-2} \log \delta^{-1} \) has been put into the term \( O(\cdot) \) for simplicity. As we know, the Hamiltonian Recovery Algorithm takes \( O(n^2) \) time. After removing the Hamiltonian Recovery procedure, the total running time of Sub Algorithm II is still \( O(n^3 \cdot t) \); thus if approximating \( \text{ham}(A_C) \) is the only purpose, outputting the Hamiltonian cycle is the by-product of Sub Algorithm II.

If the digraph \( G \) is \( \alpha \)-dense, \( \alpha > .5 \), an important result given by Huber [10] is

\[
\text{Br}(C)/\text{per}(C) = O(n^{-5+5/(2\alpha-1)}).
\]

Note that

\[
\frac{\text{per}(C)}{\text{ham}(C)} = \frac{\prod_{i=1}^{n} (X(i, i)Y(i, i)Z(i, i)) \text{per}(A_C^\epsilon)}{\prod_{i=1}^{n} (X(i, i)Y(i, i)Z(i, i)) \text{ham}(A_C^\epsilon)} = \frac{\text{per}(A_C^\epsilon)}{\text{ham}(A_C^\epsilon)}.
\]

If the digraph \( G \) is at least \( .5n \)-dense, then changing any zeros in \( A_C \) to \( \epsilon n^{-2} \) increases \( \text{per}(A_C) \) by at most a factor of \( 1 + \epsilon \) [11]. Then

\[
\frac{\text{per}(C)}{\text{ham}(C)} = \frac{\text{per}(A_C^\epsilon)}{\text{ham}(A_C^\epsilon)} \leq \frac{(1 + \epsilon/3) \text{per}(A_C)}{\text{ham}(A_C)} = O\left( \frac{\text{per}(A_C)}{\text{ham}(A_C)} \right).
\]

Hence, the total running time of our algorithm is

\[
O\left( n^4 \log n + n^2 \frac{\text{Br}(C)}{\text{ham}(C)} \right) = O\left( n^4 \log n + n^2 \frac{\text{Br}(C) \text{per}(C)}{\text{per}(C) \text{ham}(C)} \right)
\]

\[
= O\left( n^4 \log n + n^{15+5/(2\alpha-1)} \frac{\text{per}(A_C)}{\text{ham}(A_C)} \right).
\]

Now we present a combinatorial argument on the bound of \( \frac{\text{per}(A_C)}{\text{ham}(A_C)} \) (recall that \( A_C \) is a 0–1 matrix and that all the edge weights of \( G \) equal 1). The methodology is analogous to the approach for undirected graphs given by Dyer et al. [6].

**Lemma 8** [6]. Let \( n \) be a natural number and \( \beta \) a positive number. Let \( k_0 = \max(\lfloor \beta \log n \rfloor, 1) \) and \( g(k) = n^k k! (\beta \log n)^{-k} \), and define

\[
f(k) = \begin{cases} 
g(k), & k \leq k_0 
g(k_0), & k > k_0. \end{cases}
\]

Then \( f(k - 1) \geq (\beta \log n) k^{-1} f(k) \); and \( f(k) \geq 1 \) for any \( k \).

**Proof.** If \( k \leq k_0, f(k - 1) = g(k - 1) = (\beta \log n) k^{-1} g(k) = (\beta \log n) k^{-1} f(k) \).

If \( k > k_0 \), then \( \beta \log n / k \leq 1 \). Hence

\[
f(k - 1) = g(k_0) \geq (\beta \log n) k^{-1} g(k_0) = (\beta \log n) k^{-1} f(k).
\]

Thus \( f(k) \geq f(k_0) \), and we have

\[
\frac{1}{f(k)} \leq \frac{1}{f(k_0)} \leq \frac{(\beta \log n)^{k_0}}{n^\delta (k_0)!} \leq N^{-\delta} \sum_{k=0}^{\infty} \frac{(\beta \log n)^k}{(k)!} \leq n^{-\delta} e^{\beta \log n} = 1.
\]

**Theorem 9.** Suppose that \( \alpha \in (0.75, 1] \). Let \( G \) be an \( \alpha \)-dense digraph and \( F_k \) the set of 1-factors in \( G \) containing exactly \( k \) cycles, \( 1 \leq k \leq \lfloor n/2 \rfloor \). Note that \( F_1 \) is the set of Hamiltonian cycles in \( G \). Let \( F = \bigcup F_k \). Then

\[
\frac{|F|}{|F_1|} = O(n^{1+1/(2\alpha-1.5)}).
\]

With this theorem, we prove the main result of this paper **Theorem M.**

**Proof of Theorem M.** By **Theorem 9**, since \( |F|/|F_1| = \text{per}(A_C)/\text{ham}(A_C) \), and noting (1), **Theorem M** follows immediately. □

Now we proceed to prove **Theorem 9.**

**Proof of Theorem 9.** We construct a new weighted digraph \( \Psi = (F, K) \), \( K \) is defined as follows.

\[
K = ((E, E') : E \in F_k, E' \in F_{k'}, k' < k \text{ and } E \oplus E' \cong \overline{C_4}),
\]

where \( \overline{C_4} \) is a graph with four vertices and four edges, in which two vertices have indegrees 2 and outdegrees zero, and the other two vertices have indegrees zero and outdegrees 2 (see **Fig. 1**). The four edges belong to \( E \) and \( E' \) alternatively. To
avoid confusion with vertices and edges in $G$, we call the elements of $F$ and $K$ in $\Psi$ the nodes and arcs, respectively. Observe also that if $(E, E') \in K$ is an arc of $\Psi$, $E'$ can be obtained from $E$ by deleting two edges and adding two others, and that this operation can decrease the number of cycles by 1 (see Fig. 2). Hence every arc $(E, E')$ is directed from a node $E$ in some $F_k$ to a node $E'$ in $F_{k-1}$.

The proof strategy is to define a positive weight function $w$ on the arc set $K$ such that the total weight of arcs leaving each node $E \in F/F_1$ is at least 1 greater than the total weight of arcs entering $E$. Denote $w^+(S)$ and $w^-(S)$ to be the total weight leaving and entering a node set $S$ in $\Psi$, respectively; the strategy ensures that

$$w^-(F_k) + |F_k| = \sum_{E \in F_k} (w^-(E) + 1) \leq \sum_{E \in F_k} w^+(E) = w^+(F_k), \quad k \geq 2.$$ 

Hence,

$$w^-(F_1) = w^+(F_2) = \sum_{k \geq 2} (w^+(F_k) - w^-(F_k)) \geq \sum_{k \geq 2} |F_k| = |F/F_1|.$$ 

Let $g = \max_{E \in F_1} w^-(E)$. Since $w^-(F_1) = \sum_{E \in F_1} w^-(E) \leq g |F_1|$, $|F|/|F_1| \leq g + 1$.

The weight function $w : K \to R^+$ is defined as follows. For any arc $(E, E')$ with $E' \in F_k$, we know that $E'$ is obtained by coalescing two cycles of $E$, and suppose that the lengths of these two cycles are $l_1$ and $l_2$; then define $w(E, E') = (l_1^{-1} + l_2^{-1}) f(k)$, where $f(k)$ is defined as in Lemma 8. Then we have the following two claims.

**Claim 1.** For any $E \in F_k$, $k \geq 2$, $w^+(E) \geq (2\alpha - 1.5)n\beta f(k) \log n + 2$.

**Claim 2.** For any $E \in F_k$, $k \geq 1$, $w^-(E) \leq n \log nf(k)$.

By these two claims, set $\beta = 1/(2\alpha - 1.5)$. Then, for $E \in F_k$, $k \geq 2$, we have $w^+(E) - w^-(E) \geq 2 \geq 1$ and $g = \max_{E \in F_1} w^-(E) \leq n \log nf(1) \leq (2\alpha - 1.5)n^{1+1/(2\alpha - 1.5)}$. Hence $|F|/|F_1| \leq g + 1 = O(n^{1+1/(2\alpha - 1.5)})$, which completes the proof. $\Box$

**Proof of Claim 1.** Let $E \in F_k$ be a 1-factor with $k$ cycles $\gamma_1, \ldots, \gamma_k$, of lengths $n_1, n_2, \ldots, n_k$, $k \geq 2$. We proceed to bound $w^+(E)$. To show the lower bound of $w^+(E)$, we need to count the number of arcs leaving $E$. Suppose $(E, E')$ to be such an arc. Let $E = E \oplus E', E' \in F_{k-1}$, be the form $(x, x', y, y')$, where $(x, x'), (y, y') \in E$ and $(y, x') \in E'$.

First, we estimate the number of $C_4$-type cycles $\gamma$ for which $(x, x')$ is contained in a particular cycle $\gamma_i \in E$. We say that $\gamma$ is rooted at $\gamma_i$. Assume, for a moment, that the vertices $x, x'$ are chosen. There are at least $an - (n_i - 1)$ ways to extend the path first to $y$ and then to $y'$ since the indegree of $x'$ is at least $\alpha n$. Denote by $Y'$ the set of all vertices $y'$ that are reachable. Recall that $N^+(x, G)$ is the set of neighbors $x$ points to. Thus the number of ways of completing a $C_4$-type cycle $(x, x', y, y')$ is at least

$$|N^+(x, G)| + |Y'| - n \leq \alpha n + (\alpha n - (n_i - 1)) - n = 2\alpha n - n_i - n + 1.$$ 

Hence the total number of $C_4$-type cycles rooted at $\gamma_i$ is at least $n_i(2\alpha n - n_i - n + 1)$.
We are now poised to bound \( w^+(E) \). Each arc \((E, E')\) defined by a \( C_4 \)-type \( \gamma \) rooted at \( \gamma_l \) has weight at least \( n_i^{-1}f(k - 1) \), which, by Lemma 8, is bounded below by \((\beta \log n)(kn_i)^{-1}f(k)\). Thus
\[
\begin{align*}
   w^+(E) &= \sum_{E' \in \Gamma} w(E, E') \\
   &\geq \sum_{i=1}^k n_i (2\alpha n - n_l - n + 1)n_i^{-1}f(k - 1) \\
   &= \sum_{i=1}^k (2\alpha n - n_l - n)f(k - 1) + kf(k - 1) \\
   &\geq (2\alpha kn - n - kn_2)(\beta \log n)k^{-1}f(k) + kf(k - 1) \\
   &= (2\alpha - 1/k - 1)(\beta \log n)f(k)n + kf(k - 1) \\
   &\geq (2\alpha - 1.5)(\beta \log n)f(k)n + 2.
\end{align*}
\]

For the first inequality, it seems that we have overcounted the weight. We explain the reason. When \((x, x')\) is rooted at \( \gamma_l \) and \((y, y')\) lies in some \( \gamma_n \), if we extend \((x, x')\) to \((y, y')\) to complete a \( C_4 \)-type \( \gamma \), the contribution to the weight is only \( n_i^{-1}f(k - 1) \) in the above inequality. Similarly, when \((x, x')\) is rooted at the same position as \((y, y')\) in \( \gamma_l \) and \((y, y')\) lies in the same position as \((x, x')\) in \( \gamma_n \), the contribution to the weight is \( n_i^{-1}f(k - 1) \). Adding these two weights, \((n_i^{-1} + n_l^{-1})f(k - 1)\) is exactly \( w(E, E') \) needed to be considered by the definition of \( w \), where \( E \oplus E' = \gamma' \). Hence, though each \( C_4 \) cycle is counted twice, the weight is not. The last inequality follows immediately from \( k \geq 2 \) and \( f(k - 1) \geq 1 \).

**Proof of Claim 2.** For each \( E \in F_k \), we now proceed to bound \( w^-(E) \). Let \((E', E)\) be an arc in \( K \). It is straightforward to verify that the \( C_4 \)-type \( \gamma \) = \((x, x', y, y')\) = \((E, E') \oplus \gamma\) must contain two edges \((x, x')\) and \((y, y')\) from a single \( \gamma_l \) of \( E \), and \((x, x'), (x, y') \in E' \). Removing these two edges from \( \gamma_l \) leaves a double of simple paths of lengths \( p - 1 \) and \( q - 1 \), where \( p, q \geq 2 \). For the case \( p \neq q \) there are at most \( n_i \) ways such that \( \gamma_l \oplus \gamma \) is a pair of cycles with length \( p \) and \( q \), and \( n_i/2 \) ways such that \( \gamma_l \oplus \gamma \) is a pair of cycles with length \( p \) and \( q \) for the case \( p = q \). Note that both cases happen when \( \gamma_l \) is contained in a complete subgraph of \( G \) or \( G \) is a complete digraph (a complete digraph is defined as such a digraph in which any two distinct vertices have edges pointing to each other). Hence
\[
\begin{align*}
w^-(E) &= \sum_{E' \in \Gamma} w(E', E) \\
   &\leq \sum_{i=1}^k nf(k) \sum_{p \geq 2, q \geq 2, p + q = n_i} \left( \frac{1}{p} + \frac{1}{q} \right) + \sum_{i=1}^k nf(k) \sum_{p \geq 2, p + q = n_i} \left( \frac{1}{p} + \frac{1}{q} \right) \\
   &\leq \frac{1}{2} \sum_{i=1}^k nf(k) \sum_{p \geq 2} \left( \frac{1}{p} + \frac{1}{q} \right) \\
   &= \frac{1}{2} \sum_{i=1}^k nf(k) \sum_{p \geq 2} \left( \frac{1}{p} + \frac{1}{n_i - p} \right) \\
   &= \frac{1}{2} \sum_{i=1}^k nf(k) \sum_{p \geq 2} \left( \frac{1}{p} \right) \\
   &\leq \sum_{i=1}^k nf(k) \log n_i \\
   &\leq n \log nf(k). \quad \square
\end{align*}
\]

**4.2. Hardness of counting Hamiltonian cycles in dense digraphs**

Notation related to undirected graphs only appears in this subsection and the notation related to digraphs is the same as that in the previous sections. Our reduction comes from the undirected graph; hence notation for undirected graphs is needed. Let \( G \) be a simple undirected graph with vertices \( \{1, 2, \ldots, n\} \), where \( n \geq 3 \). The definition of a Hamiltonian cycle of an undirected graph is a closed undirected path that visits each vertex once and only once. We use the notation \( m_1, m_2, \ldots, m_n, m_1 \) to denote a Hamiltonian cycle in an undirected graph (recall that \( m_1, m_2, \ldots, m_n, m_1 \) denotes a Hamiltonian cycle in digraphs). The degree of a vertex in an undirected graph \( G \) is defined as the number of its neighbors. Let
#HC and #DHC be the problem of counting the number of Hamiltonian cycles in undirected and directed graphs, respectively. Now define a symmetric digraph $G'$ corresponding to an undirected graph $G$ by replacing each edge $(i, j)$ of $G$ with two directed edges $(i, j)$ and $(j, i)$. Let $H_C$ and $H_G$ denote the set of the Hamiltonian cycles in $G$ and $G'$, respectively. Let $P(H_C)$ denote the power set of $H_C$. We will prove that the number of Hamiltonian cycles in an undirected graph equals half the number of Hamiltonian cycles in its corresponding symmetric digraph.

**Lemma 10** ([6]). #HC is #P-Complete, even when restricted to graphs $G$ of minimum degree at least $(1 - \varepsilon)n$, for any $\varepsilon > 0$.

**Lemma 11.** Let $H = m_1 \cdots m_n m_1$ be a Hamiltonian cycle in $H_C$. Then there are at least two Hamiltonian cycles $(m_1, \ldots, m_n, m_1)$ and $(m_1, m_n, \ldots, m_1)$ in $H_C$. Define a map $\phi$ from $H_C$ to $P(H_G')$ as follows:

$$\phi(H) = \{(m_1, \ldots, m_n, m_1), (m_1, m_n, \ldots, m_1)\}.$$  

Let $\text{Im} \phi$ denote the image set of the map $\phi$, and let $H' = m_1' \cdots m_n' m_1'$ be a different Hamiltonian cycle from $H$ in $H_C$. Then

$$\phi(H) \cap \phi(H') = \emptyset \quad \text{and} \quad \text{Im} \phi = H_G'.$$

**Proof.** Due to the symmetry of the digraph $G'$, and noting that $n \geq 3$, for any Hamiltonian cycle $(m_1, \ldots, m_n, m_1)$ in $H_G'$, there must be a different Hamiltonian cycle $(m_1, m_n, \ldots, m_1)$ in $H_G'$. These two Hamiltonian cycles obviously have a pre-image, the Hamiltonian cycle $m_1 \cdots m_n m_1$ in $H_C$. Note that $(m_1, \ldots, m_n, m_1)$ is in $\phi(m_1 \cdots m_n m_1)$. Hence, $\text{Im} \phi \supseteq H_G'$. Obviously, $\text{Im} \phi \subseteq H_G'$. Therefore

$$\text{Im} \phi = H_G'.$$

Suppose that there are two different Hamiltonian cycles $H = m_1 \cdots m_n m_1$ and $H' = m_1' \cdots m_n' m_1'$ in $H_C$. Let $N_H(m_i)$ denote two neighbor vertices of vertex $m_i$ in $H$. $H$ and $H'$ are different if and only if there exists a vertex $m_i = \{m_i'\}$ such that $N_H(m_i) \neq N_H(m_i')$. Hence, $(m_1, \ldots, m_n, m_1)$ is different from $(m_1', \ldots, m_n', m_1')$ and $(m_1', m_n', \ldots, m_2', m_1')$; that is, $(m_1, \ldots, m_n, m_1) \notin \phi(H')$. Similarly, $(m_1, m_n, \ldots, m_1) \notin \phi(H')$. Hence $\phi(H) \cap \phi(H') = \emptyset$. □

**Theorem 12.** #DHC is #P-Complete, even when the digraph is $(1 - \gamma)n$ dense, $0 < \gamma < .5$.

**Proof.** Lemma 11 shows that the number of Hamiltonian cycles in an undirected graph is half the number of Hamiltonian cycles in its corresponding symmetric digraph. Hence, by Lemma 10, #DHC in $(1 - \gamma)n$ dense digraphs is #P-Complete, for any $0 < \gamma < .5$. □

5. Conclusions and discussion

The results in this paper show that, for relatively dense digraphs, approximating the number of Hamiltonian cycles or generating weighted Hamiltonian cycles exactly from their correct distribution can be accomplished in $O(n^{(2.5+5(2a-1)+1)/(2a-1.5)})$ time. This is an improvement in running time by a factor of $n^{1.5}(\log n)^4$ for 0.85n dense digraphs. Counting the number of Hamiltonian cycles in such digraphs is shown to be #P-Complete.

Estimating the Hamilton of a 0–1 matrix to within a factor of $1 + \varepsilon$ with probability at least $1 - \delta$, the running time is

$$O(n^{2.5+5(2a-1)+1}/(2a-1.5) - 2 \log(\delta^{-1})).$$

It is known [2] that 0.5n dense digraphs contain Hamiltonian cycles. Our algorithm presented in this paper is shown to be an FPRAS for 0.75n dense problems. Hence a gap still remains. We can extend the definition $C_4$ in the proof of Theorem 9, as shown by Fig. 1. Similarly, that can also be done to $C_6, C_8$. However, it seems unlikely to obtain any better bounds than that by $C_4$ in this way. This gap is left open here.

**Acknowledgement**

The second author’s research is supported by the National Science Foundation of China 10871115.

**References**


