International Journal of Combinatorial Optimization Problems and Informatics, 14(1), Jan-April 2023, 11-18. ISSN: 2007-1558.

www.editada.org

A Novel Method for Counting Independent Sets in a Grid Graph

Guillermo De Ita Luna, Mireya Tovar Vidal, Beatríz Bernabé Loranca
[1] Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación
deitaluna63@gmail.com, mireya.tovar@correo.buap.mx, beatriz.bernabe@gmail.com

Abstract. The problem of counting independent sets of a graph G is not only mathematically relevant and interesting, but also has many applications in physics, mathematics, and theoretical computer science. In this paper, a novel method for counting independent sets on grid-like structures is presented. It starts by explaining the recurrences used by the method to count independent sets on basic topologies of graphs. The method is extended to process grid-like structures of quadratic faces. The proposal has a lower time complexity than the required on the leading and current method based on the transfer matrix for counting independent sets on grids.
Keywords: Counting independent sets, Grid graph, Transfer matrix, Fibonacci recurrence.

Article Info
Received Sep 26, 2022
Accepted Jan 11, 2023

1 Introduction

Counting has become an important area in mathematics, as well as in computer science, even though it has received less attention than decision problems. This has caused less knowledge about the complexity of counting problems compared to the study of the complexity of decision problems. From the computational point of view, the counting of independent sets of a graph is a determining factor to establish the border between efficient counting and intractable counting procedures. Currently, there are few graph counting problems that can be solved in polynomial time.

In [1] it is shown that independent sets counting in graphs of degree 4 is a problem of the complexity class #P-Complete. Greenhill refined this work by showing that to count independent sets on graphs of degree 3 or regular 3-graphs is in the complexity class #P-Complete [2]. An important line of research is to determine the type of graphs where to count independent sets can be done in polynomial time.

The Markov chain described in [3] is one of the first approximations independent set counting algorithms. Markov chain algorithm application in a Monte Carlo system has achieved a good approximation, in polynomial time, over the number of independent sets in a graph G, especially for graphs with a maximum degree of four [4]. Many variants of the Monte Carlo algorithm have been developed (see [3, 4, 5, 6,7]) these approximation techniques fail in graphs with degree 6 or greater, but the case of the degree-5 graph still remains open [4].

There is a large amount of literature focused on counting structures on grid graphs, such as: counting spanning trees, Hamiltonian cycles, independent sets, or acyclic orientations, as well as coloring counting [8,9,10,11]. Dahllöf [12] designed a method for counting models in Boolean formulas (which would serve to count independent sets on monotone formulas), and whose algorithm is bounded above in the worst case by $O(1.3247^n)$ where n is the number of variables (vertices) of the formula. While in [13] a linear-time algorithm for counting independent sets for chordal graphs was developed. In general, the study of the number of independent sets on grid structures is closely related to the ``hard-square model'' used in statistical physics and, of particular interest is the so-called hard-square entropy constant [14].

In [8] the number of independent sets of a grid graph Gm,n (m rows and n columns, see figure 1 that illustrates a 4 x 6 grid) is calculated, using the transfer matrix method. While Euler presented the generating functions associated with counting the number of independent sets on the grid [9]. Euler also considered the problem of counting maximal independent sets on a grid. However, the application of the transfer matrix method to count the number of independent sets in a Gm,n grid has exponential time complexity over both dimensions (m and n).

© Editorial Académica Dragón Azteca S. de R.L. de C.V. (EDITADA.ORG), All rights reserved.

In statistical mechanics, counting independent sets is interpreted as counting the number of different ways of putting particles in square lattices, such that no two particles can be in the same place or at adjacent points on the squares [15]. The efficient encoding schemes in data storage are other type of application of the counting independent sets on grids [16].

In this article, we present different efficient algorithms to count independent sets on basic graph topologies such as paths, cycles, and trees. We combine those algorithms in order to process grid graphs. Our proposal dramatically reduces the time complexity required to count independent sets on grids with respect to the classical transfer matrix method.

The document is structured as follows: Section 2 presents the notation used to this research. Section 3 shows basic topologies of graphs (paths, cycles and trees) where to count independent sets is carried out efficiently. Section 4 shows how to count independent sets on grid-like structures $G_{m,n}$. Finally, section 5 contains the conclusion of the work.

2 Notation

Let $G=(V,E)$ be an undirected graph with a set of vertices V and a set of edges E. Two vertices v and w are adjacent if there is an edge $\{v,w\} \in E$ connecting them. Sometimes, we denote an edge $\{v,w\} \in E$ in abbreviated form as vw.

The neighborhood of $x \in V$ is the set $N(x)=\{y \in V:\{x, y\} \in E\}$, and its closed neighborhood is $N(x) \cup \{x\}$, which is denoted by $N[x]$. The cardinality of a set A is denoted $|A|$. The degree of a vertex x in the graph G, denoted by $\delta(x)$, that is $|N(x)|$, and the degree of the graph G is $\Delta(G) = max\{ \delta(x): x \in V\}$. The neighborhood size of x, $\delta(N(x))$, is $\delta(N(x))= \Sigma_{y \in N(x)} \delta y$.

A path between two vertices v and w is a sequence of edges: $v_0v_1,\ v_1v_2, \ldots, v_{n-1}v_n$ such that $v=v_0$ and $w=v_n$, and $v_kv_{k+1} \in E$, for $0 \leq k < n$. The length of the path is the number of edges (n) in it. A simple path is a path where $v_0v_1,\ v_1v_2, \ldots, v_{n-1}v_n$ are all distinct. A simple cycle is a simple non-empty path, where the first and last vertices are identical. An acyclic graph is a graph that does not contain cycles. We denote by P_{n-1}, C_n, and T_n to the simple path, the simple cycle, and the tree, respectively, all of them containing n vertices.

For a graph $G=(V,E)$, $S \subseteq V$ is an independent set of G, if every $v_1, v_2 \in S$, $\{v_1, v_2\} \notin E$. $I(G)$ denotes the set of all independent sets of G. An independent set $S \in I(G)$ is "maximal" if it is not a subset of a larger independent set, and it is "maximum" if it has the largest size among all independent sets in $I(G)$.

The corresponding counting problem on independent sets, denoted by $i(G)$, consists of counting the number of independent sets of a graph G. The computation of $i(G)$ is a #P-complete problem for graphs G, where $\Delta(G) \geq 3$ [1,3]. There are several polynomial procedures to compute $i(G)$ when $\Delta(G) \leq 2$ [7,12,17]. All of them are methods of linear complexity with respect to the time. An efficient procedure to calculate $i(G)$ is presented below, based on the topology of the graph G.

Counting problems are not only mathematically interesting, but they also arise in many applications. Regarding hard counting problems, the computation of the number of independent sets of a graph G has been a key in determining the frontier between efficient counting and intractable counting algorithms.

3 Basic topologies for efficient counting of independent sets

Let $i(G)= \Pi^{k}_{j=1} i(G_j)$ where G_j, $j=1, 2, \ldots, k$ are the connected components of G [8]. The total complexity of the computation time is expressed by $T(i(G))$, and it is calculated based on the maximum rule: $T(i(G))=max\{T(i(G_j)): G_j$ *is a connected component of* $G\}$.

Case A
We will consider a graph $G=(V, E)$ which consists of a simple sequence of vertices (path), $V=\{1,\ldots, n\}$ and there exist edges such that $a_i=\{i, i+1\}$, $i= 1,\ldots, n-1$ for each pair of sequential vertices.

We associate to each vertex $v_i \in V$ the pair (α_i,β_i), where α_i expresses the number of sets in $I(G_i)$ where the node v_i does not appear and β_i the number of sets in $I(G_i)$ where the node vi appears, in this way: $i(Gi)= \alpha_i + \beta_i$. The first pair (α_i,β_i) is (1,1) since on the induced subgraph $G_1 =\{v_1\}$, $I(G_1)=\{\varnothing,\{v_1\}\}$.

If we know the value of (α_i,β_i) for any $i<n$, and as the next induced subgraph G_{i+1} is built from G_i adding the vertex v_{i+1} and the edge $\{v_i, v_{i+1}\}$, it is not hard to see that the pair $(\alpha_{i+1},\beta_{i+1})$ is built from (α_i,β_i) by applying the following recurrence equation, which we will call Fibonacci recurrence:

$$\alpha_{i+1} = \alpha_i + \beta_i \;\; ; \;\; \beta_{i+1} = \alpha_i \qquad (1)$$

The series (α_i,β_i), $i=1,\ldots,n$ based on the application of recurrence (1), allows us to calculate $i(G_i)= \alpha_i + \beta_i$ for $i= 1,\ldots, n$. Then, the computation of $i(G)$ is based on the incremental calculation of $i(G_i)$, i=1, …,n. If a linear search is performed on a sequential graph G starting at one end point v_1 and moving to its incident vertices while applying recursion (1), in linear time based on the number of vertices n, $i(P_n)=i(G_n)= \alpha_n + \beta_n = F_{n+2}$ is obtained, where F_n is the nth-Fibonacci number.

If we want to carry out the process of counting independent sets on any path in a graph G, it is necessary to use computing threads or just threads. A computing thread is a sequence of pairs $\alpha_n + \beta_n$, i=1, …,n used for computing the number of independent sets on a path with n edges.

Case B

In case of having a tree-type graph G, it will be traversed based on a depth search, considering the root node as any vertex $v \in V$, and where v will be the initial node of the depth search. The pair associated with the node v ($v \in G$) is denoted by (α_v,β_v). While traversing the tree in post-order, $i(G)$ will be computed. The following considerations will be taken to calculate $i(G)$ for a tree type graph.

- Traversing G in post-order.
  When visiting a node $v \in G$ the following conditions are considered
- $(\alpha_v,\beta_v)=(1,1)$ if v is a leaf node in G.
- If v is a parent-node with a list of child-nodes $u_1,\ldots,u_k$, Fibonacci recursion will be applied to all these child-nodes (α_v,β_v) visited with $j=1, \ldots, k$ and apply $\alpha_v = \Pi^k_{j=1} \alpha_{vj}$ and $\beta_v = \Pi^k_{j=1} \beta_{vj}$. This consideration includes the case when v has only one child.
- If v is a root node of G then $(\alpha_v +\beta_v)$.

The above rules allow counting independent sets of G in order $O(n+m)$, which is the necessary time for traversing a tree structure in post-order.

Case C

Another particular case is when $G = (V, E)$, $n =m = |V| = |E|$ is a simple cycle. We can consider the cycle as a path G' of m vertices plus an additional edge that links the final vertex of the path with the initial vertex, edge $c_m=\{v_m, v_1\}$. We call to this edge $c_m=\{v_m, v_1\}$ a frond edge.

In order to calculate the number of independent sets on a simple cycle, it will be necessary to use two threads or lines of computation, one for the calculation of $i(G')$ and another one to calculate $|\{S \in I(G'): v_1 \in S \wedge v_m \in S\}|$. This calculation can be performed by fixing in $I(G')$ the independent sets where v_1 appears, which is performed with a thread $(\alpha_i' +\beta_i')$, $i=1,\ldots,m$ with initial values $(\alpha_i' +\beta_i')=(0,1)$ in order to consider that there is a single independent set of $I(G')$ where v_1 appears.

Recurrence (1) is applied to calculate the new series (α_i', β_i'), $i=2,\ldots,m$ and to consider only the independent sets where v_m appears, only the value (α_m', β_m'), is considered as the final pair $(0, \beta_m')$. An example can be seen in [18].

This leads us to $i(C_m) = i(G)=i(G') - |\{S \in I(G'): v_1 \in S \wedge v_m \in S\}|= \alpha_m + \beta_m - \beta_m' = F_{m+2} - F_{m-2}$ where F_m refers to the m-th Fibonacci number, and C_m is a cycle of m vertices . It is important to note that the last operation to perform on a thread when the last edge (a frond edge) of the cycle is recognized, is:

$$\alpha_m + \beta_m - \beta_m' \quad (2)$$

The formulas obtained for cases A and B are equivalent to the formulas obtained by Arocha [19] using Fibonacci polynomials; which are defined as $F_0(x)=1$, $F_1(x)=1$ and applying recursion for $F_n(x)=F_{n-1}(x)+xF_{n-2}(x)$ with $n \geq 2$. Meanwhile, the formula obtained for $i(C_m) = F_{m+2} - F_{m-2}$ is the m-th Lucas number. Therefore, the basic topologies; paths, simple cycles, and trees are graphs where its number of independent sets can be recognized and counted in linear-time.

4 Development

Let $G = (V, E)$ be an undirected connected graph. If we apply a depth-first search on G, it produces TG, where $V(TG) = V(G)$, TG is called the tree graph of G. The edges in TG are called tree edges, whereas the edges in $E(G) \setminus E(TG)$ are called frond edges.

Let e be a frond edge in TG. The union of the path in TG between the endpoints of e with the edge e forms a simple cycle, this is called a basic (or fundamental) cycle of G with respect to TG. Each frond edge holds the maximum path contained in the basic cycle which it is part of.

Let $C = \{C_1, C_2, \ldots, C_k\}$ be the set of fundamental cycles found during a depth-first search on G. Given any pair of basic cycles C_i and C_j from C, if C_i and C_j share any edges, then they are called intersected cycles, otherwise they are called independent cycles. We say that a set of cycles is independent when any two cycles in the set are not intersected. In particular, if two cycles share edges, then they are called adjacent.

A grid graph of size $m \times n$ is a graph $G=(V,E)$ with vertex set $V=\{(i, j): 1 \leq i \leq m, 1 \leq j \leq n\}$, and edge set $E =\{((j, i),(j+1,i)) \mid 1 \leq j < m, 1 \leq i \leq n\} \cup \{((j, i),(j,i+1)) \mid 1 \leq j < m, 1 \leq i < n\}$. We will denote a grid graph $G_{m,n}$ where m is the number of rows and n is the number of columns.

Figure 1 shows a grid with m=4 and n=6. In the figure you can see a proposed path over the vertices of the grid. The start of the path is vertex 11. When it encounters a frond edge, it is denoted by a curved arrow, with the arrowhead indicating the vertex where the cycle closes. The arrow with a crossed line indicates the end of the path.

For example, according to the notation, it is inferred that the curved arrow indicates that there will be a cycle closure by the arrow that goes from 12 to 11. The cycle closures are carried out once all the edges of the cycle are visited. As can be seen in figure 1, the first cycle that is closed is through the edge {32,31}.

The classical method for computing the number of independent sets on grid graphs is based on the transfer matrix method [8,9]. The transfer matrix method consists of building an initial matrix of F_{m+2} rows and F_{m+2} columns that are indexed by $(m + 1)$-vectors of zeros and ones.

Let us consider S as an independent set of a grid graph $G_{m,n}$ with m rows and n columns. Let Cm be the set of all $(m + 1)$-vectors $\boldsymbol{v}$ of 0's and 1's without two consecutive 1's, in which a 1 indicates that the vertex is in S and a 0 indicates that the vertex is not in S. The number of these vectors is F_{m+2}, the $(m + 2)$-th Fibonacci number. Let Tm be an $F_{m+2} \times F_{m+2}$ symmetric matrix of 0's and 1's whose rows and columns are indexed by the vectors of Cm.

The condition that vectors $\boldsymbol{u}$, $\boldsymbol{v}$ in Cm are possible consecutive pair of columns in an independent set of $G_{m,n}$ is simply that they meet the condition of having no 1's in a common position, i.e., that $\boldsymbol{u} \bullet \boldsymbol{v} = 0$ in the sense of the usual dot product of vectors over the reals.
The entry of Tm in position $(\boldsymbol{u}, \boldsymbol{v})$ is 1, if the vectors $\boldsymbol{u}$, $\boldsymbol{v}$ are orthogonal; otherwise, it is 0. Tm is called the transfer matrix for $G_{m,n}$. Then, Tm has $F_{m+2} \bullet F_{m+2}$ inputs. The number of independent sets of a grid graph $G_{m,n}$ is obtained by the sum of all entries of the n-th power matrix T^n_m, i.e., $i(G_{m,n}) = \mathbf{1}^t T^n_m \mathbf{1}$, where $\mathbf{1}$ is the (F_{m+2})-vector whose entries are all 1's.

Only the construction of the initial matrix Tm involves the order of $O((F_{m+2})^2 * (m+1))$ dot products of m + 1-vectors over the reals. Afterwards, the computation of T^n_m implicates an order of $O((F_{m+2})^{3n})$ integer multiplications, or an order of $O(((1.618)^{(m+2)*(3n)})$ if we consider the asymptotic behavior of the Fibonacci numbers and the approximation to the golden ratio

of (1.618). This last complexity could be reduced to $O((1.618)^{(m+2)*(2.81n)})$ integer multiplications, when the Strassen matrix-multiplication algorithm is applied. In any case, the application of the transfer matrix method for computing $i(G_{m,n})$ has a time-complexity of order $O((F_{m+2})^2 * (m+1) + ((F_{m+2})^{2.81n})$ integer multiplications that results in an exponential upper bound on both dimensions m and n of the grid graph $G_{m,n}$.

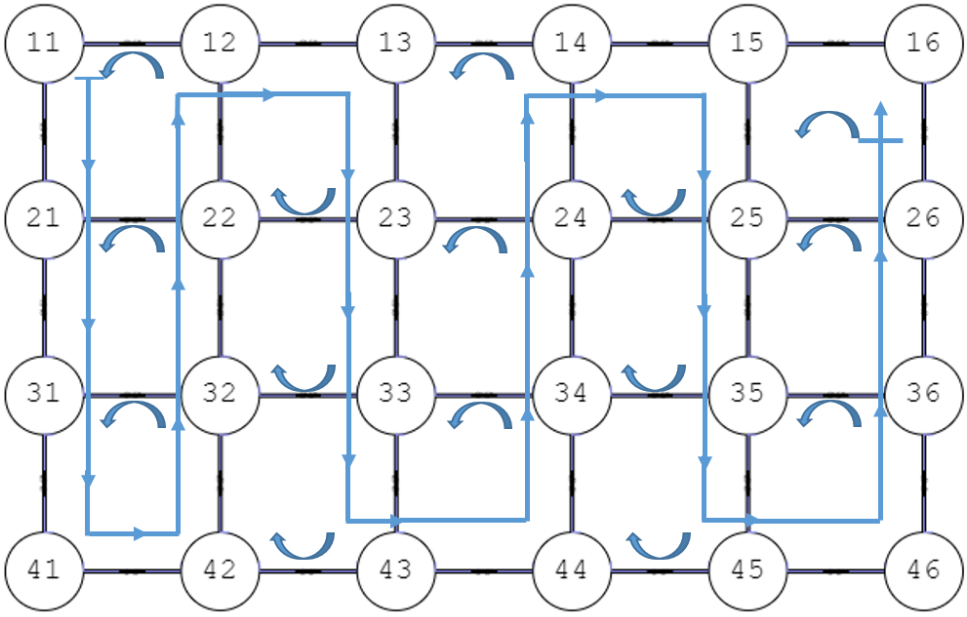

Fig. 1. A Grid graph and its path through columns.

On the other hand, our proposal is based on performing a Hamiltonian path on the grid graph. The path begins at node 11 and as we can see, it is necessary to open a computing line in addition to the main one because it is necessary to later close the cycle that started at vertex 11. Each embedded cycle will open new lines of computation, which will happen during the path when a vertex that indicates the beginning of a cycle is visited, except for vertices where the path makes a column-change. When new cycles are opened during the path, new computing threads are created, but at the same time, threads that would have been opened in previous vertices will be closed.

Table 1 shows the aforementioned process of opening and closing computation threads. Table 1 shows the process up to the cycle closure of the edge {23,22}, which is the time when the embedded cycle will reach the maximum number of open computation lines. It is important to take the following considerations that are carried out during the tour, such as the application of recurrence, the closing of the cycle and how the computation lines will be opened.

- At each movement from one vertex to the next, the recurrence (1) is applied.
- Each time a cycle is opened, as many computation lines are opened as are currently active with a value $\beta \neq 0$ associated with the pair (α,β) of the line.
- For a cycle closure, equation (2) will be applied, where the value of β's will be subtracted with respect to the active line with which it was opened.

Due to the nature of the growth in the number of computing lines, Table 1 is divided into subsections of the process. In each subsection, the computing lines that have already been closed will be eliminated, and then the compilation of the lines that are still open will be carried out. At the closing of the cycle of the edge {23,22}, the maximum number of computing lines is obtained, which can be seen to coincide with the Fibonacci value F_7. The results of counting independent sets shown in Table 1 have been verified using an exponential-time program that counts "Independent Sets" exhaustively.

Table 1. The number of independent sets corresponding to Figure 1

THREADS / NODES	11	21	31	41	42	32 ↶ 31	22 ↶21	12 ↶ 11
Lp	(1,1) →	(2,1) →	(3,2) →	(5,3) →	(8,5) →	(13,8) – (0,2) = (13,6) →	(19,13) – (0,3) = (19,10) →	(29,19) – (0,7) = (29,12) →
$C_{11\ p}$	(0,1) →	(1,0) →	(1,1) →	(2,1) →	(3,2) →	(5,3) – (0,1) = (5,2) →	(7,5) →	(12,7) →X
$C_{21\ P}$	----------	(0,1) →	(1,0) →	(1,1) →	(2,1) →	(3,2) →	(5,3) →X	----------
$C_{31\ p}$	----------	----------	(0,2) →	(2,0) →	(2,2) →	(4,2) →X	----------	----------
$C_{31\ 11}$	------	----------	(0,1) →	(1,0) →	(1,1) →	(2,1) →X	----------	----------
$C_{42\ P}$	-----	----------	--------	----------	(0,5) →	(5,0) →	(5,5) – (0,1) = (5,4) →	(9,5) – (0,2) = (9,3) →
$C_{42\ 11}$	----------	----------	----------	----------	(0,2) →	(2,0) →	(2,2) →	(4,2) →x
$C_{42\ 21}$	----------	----------	----------	----------	(0,1) →	(1,0) →	(1,1) →X	----------
$C_{42\ 31\ P}$	----------	----------	----------	----------	(0,2) →	(2,0) →X	----------	----------
$C_{42\ 31\ 11}$	----------	----------	----------	----------	(0,1) →	(1,0) →X	----------	----------
$C_{32\ P}$	----------	----------	----------	----------	----------	(0,6) →	(6,0) →	(6,6) – (0,2) = (6,4) →
$C_{32\ 11}$	----------	----------	----------	----------	----------	(0,2) →	(2,0) →	(2,2) →X
$C_{32\ 21}$	----------	----------	----------	----------	----------	(0,2) →	(2,0) →X	----------
$C_{22\ P}$	----------	----------	----------	----------	----------	----------	(0,10) →	(10,0) →
$C_{22\ 11\ P}$	----------	----------	----------	----------	----------	----------	(0,5) →	(5,0) →X
$C_{22\ 42\ P}$	----------	----------	----------	----------	----------	----------	(0,4) →	(4,0) →
$C_{22\ 42\ 11}$	----------	----------	----------	----------	----------	----------	(0,2) →	(2,0) →X

Table 2.1 Continuation of the number of independent sets corresponding to Figure 1

THREADS/ NODES	13	23 ↶ 22	33 ↶ 32	43 ↶ 42	44	34 ↶ 33
Lp	(41,29) →	(70,41) – (0,10) = (70,31)	(101,70) – (0,16) = (101,54) →	(155,101) – (0,29) = (155, 72) →	(227,155) →	(382, 227) – (0,54) = (382,173) →
$C_{42\ P}$	(12,9) →	(21,12) – (0,4) = (21,8)	(29,21) →	(50,29) →X	----------	----------
$C_{32\ P}$	(10,6) →	(16,10) →	(26,16) →X	----------	----------	----------
$C_{22\ P}$	(10,10) →	(20,10) →X	----------	---------	---------	----------
$C_{22\ 42\ P}$	(4,4) →	(8,4) →X	----------	----------	----------	----------
$C_{13\ P}$	(0,29) →	(29,0) →	(29,29) – (0,6) = (29,23) →	(52,29) – (0,9) = (52,20) →	(72,52) →	(124,72) – (0,23) = (124,49) →
$C_{13\ 42\ P}$	(0,9) →	(9,0) →	(9, 9) →	(18, 9) →X	----------	----------
$C_{13\ 32\ P}$	(0,6) →	(6,0) →	(6,6) →X	----------	---------	----------
$C_{13\ 22\ P}$	(0,10) →	(10,0) →X	----------	----------	---------	----------
$C_{13\ 22\ 42\ P}$	(0,4) →	(4,0) →X	--------	----------	----------	----------
$C_{23\ P}$	----------	(0,31) →	(31,0) →	(31,31) – (0,8) = (31,23) →	(54,31) →	(85,54) →
$C_{23\ 42\ P}$	----------	(0,8) →	(8,0) →	(8,8) →X	----------	-------
$C_{23\ 32\ P}$	----------	(0,10) →	(10,0) →X	----------	----------	----------
$C_{33\ P}$			(0,54) →	(54,0) →	(54,54) →	(108,54) →X
$C_{33\ 42\ P}$			(0,21) →	(21,0) →X	----------	----------
$C_{33\ 13\ P}$			(0,23) →	(23,0) →	(23,23) →	(46,23) →X
$C_{33\ 13\ 42\ P}$			(0,9) →	(9,0) →X	------	----------
$C_{44\ p}$					(0,155) →	(155,0) →
$C_{44\ 13\ P}$					(0,52) →	(52,0) →
$C_{44\ 23\ P}$					(0,31) →	(31,0) →
$C_{44\ 33\ P}$					(0,54) →	(54,0) →X
$C_{44\ 33\ 13\ P}$					(0,23) →	(23,0) →X
$C_{34\ P}$						(0, 173) →
$C_{34\ 13\ P}$						(0,49) →
$C_{34\ 23\ P}$						(0,54) →

Table 3.2 Continuation of the number of independent sets corresponding to Figure 1

THREADS/ NODES	24 ↶ 23	14 ↶ 13	15	25 ↶ 24	35 ↶ 34	45 ↶ 44
Lp	(555,382) −(0,85) = (555,297) →	(852,555) − (0,173) = (852,382) →	(1234,852) →	(2086,1234) − (0,297) = (2086,937) →	(3023,2086) − (0,470) = (3023,1616) →	(4639,3023) − (0,919) = (4639,2104) →
$C_{13\ P}$	(173,124) →	(297,173) →X	----------	---------	----------	----------
$C_{23\ P}$	(139,85) →X	---------	----------	----------	----------	----------
$C_{44\ p}$	(155,155) − (0,31) = (155,279) →	(279,155) − (0,52) = (279,103) →	(382,279) →	(661,382) − (0,124) = (661,258) →	(919,661) →	(1580,919) →X
$C_{44\ 13\ P}$	(52,52) →	(104,52) →X	----------	----------	----------	----------
$C_{44\ 23\ P}$	(31,31) →X	----------	----------	----------	----------	---------
$C_{34\ p}$	(173,0) →	(173,173) − (0,49) = (173,124) →	(297,173) →	(470,297) →	(767,470) →X	----------
$C_{34\ 13\ P}$	(49,0) →	(49,49) →X	----------	----------	---------	----------
$C_{34\ 23\ P}$	(54,0) →X	----------	----------	----------	---------	----------
$C_{24\ P}$	(0,297) →	(297,0) →	(297,297) →	(594,297) →X	----------	----------
$C_{24\ 13\ P}$	(0,124) →	(124,0) →X	----------	---------	----------	----------
$C_{24\ 44\ P}$	(0,124) →	(124,0) →	(124,124) →	(248,124) →X	----------	-------
$C_{24\ 44\ 13\ P}$	(0,52) →	(52,0) →X	----------	----------	----------	----------
$C_{15\ P}$	----------		(0,852) →	(852,0) →	(852,852) − (0,173) = (852,679) →	(1531,852) − (0,279) = (1531,573) →
$C_{15\ 44\ P}$	----------		(0,279) →	(279,0) →	(279,279) →	(558,279) →X
$C_{15\ 34\ P}$	----------		(0,173) →	(173,0) →	(173,173) →X	----------
$C_{15\ 24\ P}$	----------		(0,297) →	(297,0) →X	------	----------
$C_{15\ 24\ 44\ P}$	----------		(0,124) →	(124,0) →X	----------	----------
$C_{25\ P}$	----------			(0,937) →	(937,0) →	(937,937) − (0,258) = (937,679) →
$C_{25\ 44\ P}$	----------			(0,258) →	(258,0) →	(516,258) →X
$C_{25\ 34\ P}$	----------			(0,297) →	(297,0) →X	---------
$C_{35\ P}$	----------				(0,1616) →	(1616,0) →
$C_{35\ 44\ P}$					(0,661) →	(661,0) →X
$C_{35\ 15\ P}$					(0,679) →	(679,0) →
$C_{35\ 15\ 44\ P}$					(0,279) →	(279,0) →X

Table 4.3 Continuation of the number of independent sets corresponding to Figure 1

LÍNEAS/VÉ RTICES	46	36 ↶ 35	26 ↶ 25	16 ↶ 15	$i(G)$
Lp	(6743,4639) →	(11382,6743) − (0,1616) = (11382,5766) →	(16037,11382) − (0,2553) = (16037,13484) →	(25338,16037) − (0,5060) = (25338,11449) →	36787
$C_{15\ P}$	(2104,1531) →	(3635,2104) − (0,679) = (3635,1425) →	(5060,3635) →	(8695,5060) →X	----------
$C_{25\ P}$	(1616,937) →	(2553,1616) →	(4169,2553) →X	----------	----------
$C_{35\ p}$	(1616,1616) →	(3232,1616) →X	----------	---------	----------
$C_{35\ 15\ P}$	(679,679) →	(1358,679) →X	----------	----------	----------

The time complexity of our proposal is related to the dimensions of Table 1, which we denote by $T_{k,l}$, the table with k rows and l columns. For a grid $G_{m,n}$, the number of columns of the grid coincides with the total number of vertices that are visited during the path, which corresponds to the number of vertices in the grid, so $l = m*n$.

The number of rows k in the table is a dynamic value that changes as computing lines are opened and closed, and rows are moved in the table to avoid gaps in the table. To estimate k, we must consider what is the maximum number of lines that can be opened at any time during the calculation of $i(G_{m,n})$. The maximum number of open lines corresponds to the number of vertices in the first column plus one vertex in the second column, which is when all cycle lines are active, and no cycle has been closed. Therefore, this value corresponds to $(m+1)$ opening cycles.

Each time that a new cycle is processed, a Fibonacci growth on the number of computing threads is followed. At the start of the path, two computation lines are created, then 3, followed by 5, 8, which corresponds to a Fibonacci growth. So, after the $(m+1)$ cycle starts, we will have $k=F_{m+3}$ active computing lines, since the Fibonacci sequence started with 2 lines (so the sequence starts with F_3). Therefore, $k=F_{m+3}$.

The maximum number of cells in the Table will be $k * l = F_{m+3} * (m * n)$, which also corresponds to the order of growth of the time complexity of our process. Our algorithm then has a time complexity of order $O(F_{m+3} * (m * n)) = O((1.618)^{m+3} * (m * n))$ considering 1.618 as an approximation to the 'golden ratio', which is the constant that allows us to calculate any Fibonacci number.

Therefore, the time-complexity of our proposal for computing $i(G_{m,n})$ is dramatically inferior to the time-complexity that the transfer matrix method requires for computing $i(G_{m,n})$.

5 Conclusions

Given a grid graph $G_{m,n,}$ in order to compute $i(G_{m,n)}$, it is necessary to visit all vertex in $G_{m,n}$. The traversing is in the direction of the column if $m < n$; otherwise, we can rotate the grid and obtain that $m \leq n$ is always satisfied. Afterwards, we apply a traversing by column in order to compute $i(G_{m,n)}$. We propose a Hamiltonian path by columns on the grid graph, where the maximum number of computing lines that will be active at any time of the counting process is upper bounded by $O(F_{m+3}*(m*n))=O(F_{min\{n,m\}+3}*(m*n)) \cong O((1.1618)^{min\{n,m\}+3}*(m*n))$. The resulting time-complexity of our proposal for computing $i(G_{m,n)}$ is dramatically inferior to the time-complexity that the transfer matrix method requires for computing the same value, since the transfer matrix method has an exponential complexity in both dimensions (m and n) of the grid.

It is important to note that the type of traversing on $G_{m,n}$ is vital in order to obtain minimum time-complexity, since using a different path may increase the time-complexity of the entire development. A future work is to determine which is the optimal path to obtain a minimum time-complexity when $i(G_{m,n)}$ is being computed in a grid, even for irregular grids (grids with a different number of columns per row).

References

1. Vadhan Salil P.: The complexity of counting in sparse, regular, and planar graphs. SIAM Journal on Computing, 31, no.2, 398--427 (2001)
2. Greenhill C.: The complexity of counting colouring and independent sets in sparse graphs and hypergraphs. Computational Complexity, 52--72 (2000)
3. Luby M., Vigoda E.: Approximately counting up to four, Twenty-ninth annual Symp. On theory of computing. ACM NEW York (1997) 682--687
4. Dyer M., Greenhill C.: Some #P-completeness proofs for colouring and independent sets. Research report series, university of Leeds. (1997)
5. Dyer M., Frieze A., Jerrum M.: On counting independent sets in sparse graphs. SIAM J. Comput. 31, no. 5, 1527—1541 (2002)
6. Dyer M., Greenhill C., Corrigendum: The complexity of counting graph homomorphism. RSA: Random Strutures and Algorithms. 346—352 (2004)
7. Russ B.: Randomized Algorithms: Approximation, generation, and counting, distinguished dissertations. Springer (2001)
8. Calkin N., Wilf H.: The number of independent sets in a grid graph. SIAM Journal on Discrete Mathematics, 11, 02 (1998)
9. Reinhardt E.: The Fibonacci number of a grid graph and a new class of integer sequences. Journal of Integer Sequences, 8, 05 (2005)
10. Mordecai G., Yiu Cho L., Yajun W., and Xuerong Y.: Counting structures in grid graphs, cylinders and tori using transfer matrices: Survey and new result. In ALENEX/ANALCO, pp. 250–258, 01 (2005)
11. Guillen C., López A., and De Ita G.: Computing #2-sat of grids, grid-cylinders and grid-tori Boolean formulas. In Proc. of the 15th RCRA workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion, vol. 451. CEUR-WS.org (2008)
12. Dahllöf V., Jonsson P.: An algorithm for counting maximum weighted independent sets and its applications. Proc. Of SODA (2002) thirteenth annual
13. OkamotoY., Uno T., Uehara R.: Counting the number of independent sets in chordal graphs. Proc. of the 31st Int. WS on Graph-Theoretic concepts in computer science. WG 2005, pp. 433--444, LNCS, vol. 3787 (2005)
14. Baxter R.J.: Planar lattice gases with nearest-neighbor exclusion, Annals of Combinatorics, Springer Science and Business Media 3, (1999) 191--203
15. Zhang Z.: Merrifield-Simmons index and its entropy of the 4-8-8 lattice. J Statical Physics 154, (2014) 1113-1123
16. Roth R.M., Siegel P.H., and Wolf J.K. Efficient coding schemes for the hard-square model, IEEE Trans. Inform. Theory, 47:1166-1176, (2001)
17. Roth D.: On the hardness of approximate reasoning. Artificial intelligence 82, 273--302 (1996)
18. Zacarias F., Moyao Y., Contreras M., De Ita G., Bello P.: Combinational algorithms and learning. Montiel & Soriano, pp 33-44 (2017)
19. Arocha J.L.: Propiedades del polinomio independiente de un grafo. Revista Ciencias Matemáticas, vol. V, 103—110 (1984)