Séminaire Lotharingien de Combinatoire **54A** (2007), Article B54Ak

A DIRECT DECOMPOSITION OF 3-CONNECTED PLANAR GRAPHS

MANUEL BODIRSKY, CLEMENS GRÖPL, DANIEL JOHANNSEN, AND MIHYUN KANG

ABSTRACT. We present a decomposition strategy for c-nets, i. e., rooted 3-connected planar maps. The decomposition yields an algebraic equation for the number of c-nets with a given number of vertices and a given size of the outer face. The decomposition also leads to a deterministic and polynomial time algorithm to sample c-nets *uniformly at random*. Using rejection sampling, we can also sample isomorphism types of convex polyhedra, i.e., 3-connected planar graphs, uniformly at random.

RÉSUMÉ. Nous proposons une stratégie de décomposition pour les cartes pointées 3-connexes (c-réseaux). Cette décomposition permet d'obtenir une équation algébrique pour le nombre de c-réseaux suivant le nombre de sommets et la taille de la face extèrieure. On en déduit un algorithme de complexité en temps polynomiale pour le tirage aléatoire uniforme des c-réseaux. En utilisant une méthode à rejet, nous obtenons aussi un algorithme de tirage aléatoire uniforme pour les graphes planaires 3-connexes.

1. INTRODUCTION

Three-connected planar graphs are in a one-to-one relationship to the edge-graphs of convex polyhedra [24]. The enumeration of such graphs has a long history. Already Euler attempted to find an exact formula for the number of isomorphism types of convex polyhedra [10], which is still unknown. However, since almost all such graphs have a trivial automorphism group [3,27], and since all embeddings of such a graph are equivalent (due to Whitney; see e.g. [9]), the *asymptotic* behavior of these numbers is the same as for the number of *c-nets*, i.e., three-connected planar maps with a distinguished directed edge at the outer face. The exact and the asymptotic number of c-nets for a given number of edges was first computed by Tutte [26]. Mullin and Schellenberg [19] found exact formulas in terms of vertices and faces. The algebraic equation derived there was analyzed by Bender and Richmond in [2], who showed that the growth constant for the number of c-nets depending on the number of vertices is $16/27(17+7\sqrt{7}) \doteq 21.049042$.

Other motivations to study c-nets come from *random sampling* in theoretical computer science[1]. The only known algorithm to sample labeled planar graphs uniformly at random in polynomial time requires a sampling procedure for c-nets in its “inner loop” [4]. A sampling procedure from [1, 22, 23] for planar maps with given numbers of vertices and edges was applied for that step in [4], and the analysis shows that this is the bottleneck for the performance. Recently, the sampling procedure for c-nets was improved [13]. But still this approach applies rejection sampling, and therefore can only lead to *expected* polynomial time sampling procedures.

Key words and phrases. Random sampling, planar graphs, algorithms.

[1]In the literature often the word “generating” is used instead of “sampling”. We prefer “sampling” because it is more specific.

In this paper, we present a new decomposition strategy for the number of c-nets with a given number of vertices and a given size of the outer face. We will formulate the decomposition using the generating function for the number of c-nets. The resulting equations can be solved with the quadratic method [6, 12], and the generating function for the number of c-nets is algebraic of degree four, and therefore has an explicit description with radicals. Using the computer algebra package GFUN [21], we compute a linear differential equation with polynomial coefficients that describes the generating function. From that we get a single-parameter recurrence for its coefficients that allows to compute the number of c-nets with more than 100000 vertices within reasonable time. Following the discussion in the forthcoming book of Flajolet and Sedgewick [12] we compute the mentioned growth constant.

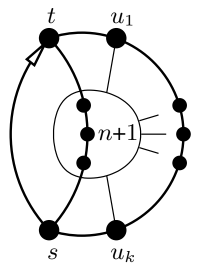
With the decomposition strategy we obtain the first *deterministic* polynomial time sampling procedure for c-nets. Together with the results in [4] we obtain the first *deterministic* polynomial time sampling procedure for labeled planar graphs. Since almost all 3-connected graphs have a trivial automorphism group [3], this can also be used in a rejection sampling procedure to sample 3-connected planar graphs in *expected* polynomial time. The algorithm uses a recursive formula for c-nets on n vertices with a specified size of the outer face. Our decomposition strategy is flexible enough to also control other parameters of c-nets, for instance the total number of edges, faces, or the degrees of root vertices, if needed. From a methodological point of view, the decomposition is interesting, since it generalizes the well-known and classical approach of Tutte to count triangulations [25]. This direct technique was never carried out for c-nets – yet it is particularly suited for the *recursive method for sampling* (an early reference is [20]; see [8, 11] for recent developments).

The fact that we can control the size of the outer face opens new applications for counting *unlabeled* planar graphs. The only approach in question to enumerate unlabeled planar graphs exploits the connectivity structure, and was already proposed in [28]. As a first step, we can use the result of the present paper to compute the number of *unlabeled rooted 2-connected* planar graphs on a given number of edges. Moreover, using the sampling procedure for c-nets with a specified size of the outer face, we obtain the first expected polynomial time sampling procedure for unlabeled 2-connected planar graphs [5]. With the sampling procedures for c-nets from [13] this is not possible.

Outline. The paper is organized as follows: We first introduce c-nets, and mention previous enumerative results. In Section 3, we describe the unique decomposition strategy for c-nets, which directly translates into equations for the generating function for the number of c-nets. In Section 4 we apply the quadratic method to derive a single algebraic equation of degree four that defines this generating function, and to derive a single parameter recurrence. Section 5 uses the decomposition to sample c-nets uniformly at random.

2. PLANAR STRUCTURES AND C-NETS

A *map* is a graph embedded in the plane. A *planar graph* is a graph that has an embedding in the plane. A graph is k*-connected* if the graph stays connected after deleting any k vertices. By Whitney's theorem (see e.g. [9]), all embeddings of 3-connected planar graphs are equivalent. A *rooted map* is a map with a distinguished directed edge st



c-net

FIGURE 1. A c-net on $n + k + 3$ vertices

(called the root edge) on the outer face. If we count rooted maps, we count them up to isomorphisms that map the outer face to the outer face and the root edge to the root edge.

A *c-net* is a rooted and 3-connected map on at least three vertices. We distinguish between *outer* vertices which lie on the outer face, and *inner* vertices which do not lie on the outer face. The outer vertices include the vertices of the root st and are labeled $s, t, u_1, \ldots, u_k$ in clockwise order starting with the root; see Figure 1.

Starting with Tutte's pioneering work [26], many classes of planar maps were enumerated. It is possible to compute the number of unrooted planar maps on m edges [18,29,30]. For *rooted* maps, the enumeration is easier. The formulas for 3-connected, 2-connected, connected, and all rooted planar maps are related via a connectivity decomposition [26]. Mulling and Schellenberg [19] used a bijection between 3-connected rooted maps, i.e., c-nets, and *quadrangulations*, which can be further decomposed, to enumerate c-nets in terms of edges and faces (by Euler's formula, one can then also control the number of vertices). The evaluation of their formula, however, involves the evaluation of a double summation. In this paper, we present a single parameter recurrence that can be computed much faster. Since the generating function is algebraic, it is straightforward to use singularity analysis (an excellent exposition of which can be found in the forthcoming book of Flajolet and Sedgewick [12]) to reproduce the asymptotic results of Bender and Richmond [2].

3. DECOMPOSITION

In this section we present a unique decomposition strategy for c-nets. Informally, the idea is to *remove* the root edge, and to describe the remaining graph in terms of smaller c-nets. Tutte [25] applied this technique successfully to *near-triangulations*, which generalize triangulations. The decomposition proposed by Tutte is simple: Either the graph without the root edge is 3-connected, or it is decomposed at its 2-cuts into 3-connected components. In either case the decomposition yields one or more smaller near-triangulations. The uniqueness of the decomposition is ensured by an important property of the simple structure of near-triangulations: The components of a decomposition at a 2-cut are independent, i.e., an arbitrary combination of near-triangulations can be composed to obtain a near-triangulation.

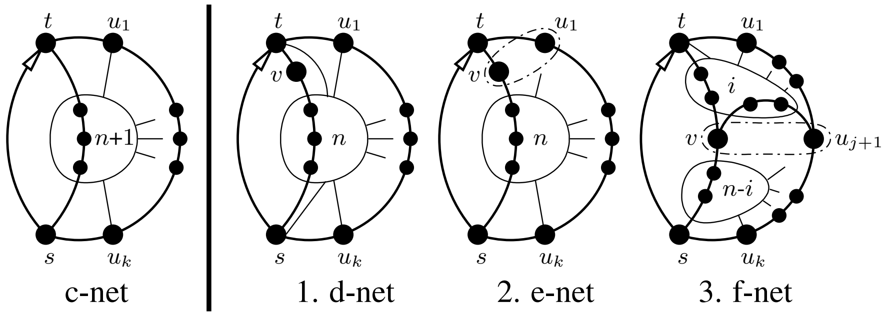


FIGURE 2. The basic case distinction: Every c-net (except K_3) is either a d-net, and an e-net, or an f-net.

The generalization of this decomposition to c-nets faces mainly two problems. First, the objects resulting from the decomposition (i.e., the removal of the root edge) are not necessarily c-nets. Second, the components induced by a 2-cut are in general not independent as described before. We solve these problems by assigning distinct generating functions to each type of component and by introducing a third case for the decomposition into dependent components. This leads us to the notions of d-nets (one 3-connected component), e-nets (there is a 2-cut that yields two dependent components) and f-nets (there is a 2-cut that yields two independent components), which are depicted in Fig. 3.

In figures, we draw the root edge st as a directed edge. Edges that are added to the graph are indicated as dotted lines. If a pair of vertices forms a 2-cut, we draw a dashed circle around the two vertices. The set of inner vertices is represented by a closed line with its size noted inside.

We formulate the decomposition in terms of generating functions. Let $c(n,k)$ be the number of all c-nets on $n+1$ inner vertices and $k+2$ outer vertices. For technical reasons, we define *double rooted* c-nets where the root can be a double edge. In particular, the outer face of a double rooted c-net is bounded by the root st and another single undirected edge. By definition of $c(n,k)$ the number of double rooted c-nets on $n+1$ inner and two outer vertices is $c(n,0)$. Since every double rooted c-net can be identified with a simple c-net by removing the undirected edge, the number of c-nets on $n+3$ vertices in total is $c(n) := c(n,0)$. Furthermore, this operation transforms k inner vertices into outer vertices, hence $c(n,0) = \sum_{k=1}^{n} c(n-k,k)$. Finally, let $C(t,u) := \sum_{n\geq 0}\sum_{k\geq 0} c(n,k)t^n u^k$ be the ordinary generating function for the number of c-nets, and let $C(t) := \sum_{n\geq 0} c(n)t^n$.

Decomposition of c-nets. If a c-net has only three vertices (s, t, and an inner vertex) then it is the K_3 with a double root and represents the only initial case of the whole decomposition. (The decomposition terminates trivially for negative values of n or k.) Now consider c-nets on at least four vertices. We distinguish three disjoint cases; they are depicted in Fig. 2.

1. After removing the root edge, the remaining graph is still three-connected.

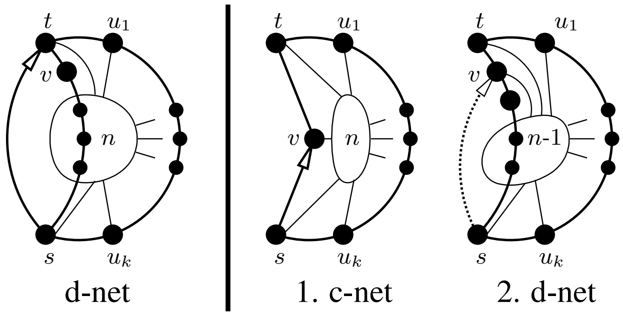


FIGURE 3. The decomposition of d-nets.

2. The vertex t is of degree three, and there is a 2-cut in the graph without the root edge. (The two remaining neighbors of t necessarily form a 2-cut in the graph without the root.)
3. The vertex t is at least of degree four, and there is a 2-cut in the graph without the root edge.

Now let $D(n, k)$, $E(n, k)$ and $F(n, k)$ be the generating functions representing the c-nets of the first, second and third case, with coefficients $d(n, k)$, $e(n, k)$ and $f(n, k)$. For convenience we call these three different kind of c-nets *d-nets*, *e-nets* and *f-nets*. Then the basic case distinction can be formulated as follows.

$$C(t, u) = 1 + D(t, u) + E(t, u) + F(t, u)\,. \tag{1}$$

Decomposition of d-nets. Let G be a d-net, i.e., G is a c-net which is 3-connected after removing the root st. The decomposition of d-nets is easy. Let v be the neighbor of t (different from s) on the inner face that contains the root st. There are two distinct cases, depicted in Fig. 3.

1. The vertex v is the only vertex on the inner face of st except s and t.
   Decomposition: Remove st and choose sv as new root edge.
   Result: A c-net with one inner vertex less and one outer vertex more than G.
2. There is at least one other vertex than v on the inner face of st except s and t.
   Decomposition: Remove st and insert sv as new root edge.
   Result: A d-net with one inner vertex less and one outer vertex more than G.

According to the case distinction the generating function $D(t, u)$ is the sum of the generating functions $C(t, u)$ and $D(t, u)$ multiplied by t for the removed inner vertex and divided by u for the additional outer vertex. From this we have to subtract $C(t, 0)$ and $D(t, 0)$ (again multiplied by t and divided by u), since the resulting graph cannot be a double rooted c-net.

$$D(t, u) = \frac{t}{u}(C(t, u) + D(t, u)) - \frac{t}{u}(C(t, 0) + D(t, 0))\,. \tag{2}$$

With exception of the initial case every c-net with a double edges root is a d-net. Hence

$$D(t, 0) = C(t, 0) - 1\,. \tag{3}$$

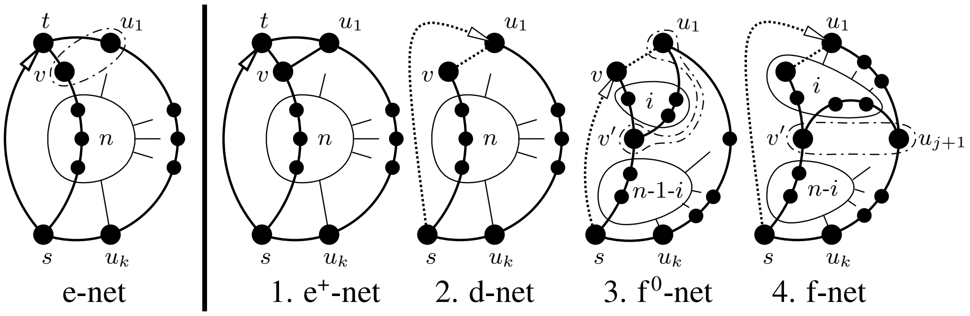


FIGURE 4. The decomposition of e-nets.

Decomposition of e-nets. Let G be an e-net, i.e., G is a c-net and t is of degree 3. The two neighbors of t apart from s are u_1 on the outer and v on the inner face and $\{v, u_1\}$ forms a 2-cut on G without st. We now introduce the last two kinds of c-nets that appear in the decomposition. *e^+-nets* (represented by $E^+(t, u)$) are defined as e-nets where the two neighbors (other than s) of t are connected by an edge, whereas *f^0-nets* (represented by $F^0(t, u)$) are defined as f-nets where u_1 has to be one of the cut vertices. In the decomposition of d-nets there are four distinct cases; they are depicted in Fig. 4.

1. There is an edge vu_1 in G.
   Result: An e+-net with the same number of vertices like G.
2. There is no edge vu_1 and G without t is 3-connected.
   Decomposition: Remove t, insert the edge vu_1 and insert su_1 as new root edge.
   Result: A d-net with one outer vertex less than G.
3. There is no edge vu_1 and G without t has a 2-cut including u_1.
   Decomposition: Remove t, insert vu_1 and insert sv as new root edge.
   Result: An f0-net with one inner vertex less than G.
4. There is no edge vu_1 and G without t has a 2-cut, where u_1 is no cut vertex.
   Decomposition: Remove t, insert vu_1 and insert su_1 as new root edge.
   Result: An f-net with one outer vertex less than G.

The decomposition of e-nets yields the following equation where the four terms correspond to the respective cases and the factors t and u account for the removed vertices.

$$E(t, u) = E^+(t, u) + u\, D(t, u) + t\, F^0(t, u) + u\, F(t, u)\,. \tag{4}$$

Decomposition of e+-nets. Next, let G be an e+-net, i.e., an e-net with an edge vu_1. Again, there are four distinct cases; they are depicted in Fig. 5.

1. The degrees of v and u_1 in G are both three.
   Decomposition: Remove t and u_1, insert the edge vu_2 (which cannot exist in G) and insert sv as new root edge.
   Result: An e-net with one inner and one outer vertex less than G.
2. The degree of v in G is three and the degree of u_1 in G is at least four.
   Decomposition: Remove t and insert sv as new root edge.
   Result: An e-net with one inner vertex less than G.

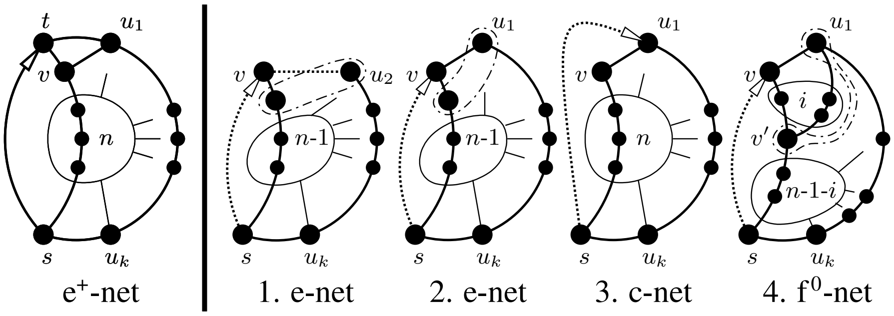


FIGURE 5. The decomposition of e⁺-nets.

3. The degree of v in G is at least four, and u_1 is not a cut-vertex of any 2-cut in G without t.
   Decomposition: Remove t and insert su_1 as new root edge.
   Result: A c-net with one outer vertex less than G.
4. The degree of v in G is at least four, and u_1 is a cut-vertex of a 2-cut in G without t.
   Decomposition: Remove t and insert sv as new root edge.
   Result: An f^0-net with one inner vertex less than G.

In the equation defining $E(t,u)$ the four terms again correspond to the respective cases and the factors t and u account for the removed vertices.

$$E^{+}(t,u) = tu\ E(t,u) + t\ E(t,u) + u\ C(t,u) + t\ F^0(t,u)\,. \tag{5}$$

Decomposition of f-nets and f^0-nets. Let G be an f-net, i.e., G is a c-net where the degree of t is at least four and which has a 2-cut after removing st. Because of planarity there exists a unique 2-cut vu_{j+1} $(0 \le j \le k-1)$ that is closest to t (see Figure 6). As introduced above, G is an f^0-net if $j = 0$. G without v and u_{j+1} has two components, one of which includes t and i inner vertices and the other includes s and $n-i$ inner vertices. Let G_t be the subgraph induced by v, u_{j+1} and the component containing t, and let G_s be the subgraph induced by v, u_{j+1} and the component containing s. Note that the edge vu_{j+1} might or might not be present in G.
Decomposition: If vu_{j+1} is not an edge of G, then insert it into both G_t and G_s. Insert tu_{j+1} as root edge into G_t. Add a new vertex t' to G_s, insert the edges st', $t'v$ and $t'u_{j+1}$, and choose st' to be the root edge of G_s.
Result: G_t is a d-net with i inner and j outer vertices. G_s is an e⁺-net with $n-i$ inner and $k-j$ outer vertices. For given parameters i and j the choice whether vu_{j+1} is an edge of G, the choice of G_t and the choice of G_s are all independent, i.e., changing any of these choices in an f-net yields a different f-net with the same parameters.

The later two choices account for the product of $D(t,u)$ and $E^{+}(t,u)$ in the equation defining $F(t,u)$ while the first choice contributes a factor of two.

$$F(t,u) = 2\ D(t,u)E^{+}(t,u)\,. \tag{6}$$

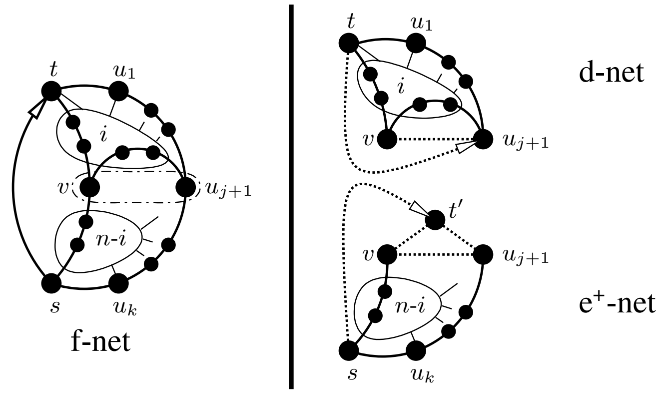


FIGURE 6. The decomposition of f-nets and f^0-nets.

The decomposition for f^0-nets is the same, and since an f^0-net is an f-net with $j = 0$, we have

$$F^0(t,u) = 2\,D(t,0)E^{+}(t,u) \tag{7}$$

4. GENERATING FUNCTIONS

We now use the equations (1)–(7) to derive an algebraic equation and an explicit description for $C(t,u)$ and for $C(t) = C(t,0)$. First, we eliminate the auxiliary generating functions $D(t,u)$, $D(t,0)$, $E(t,u)$, $E^{+}(t,u)$, $F(t,u)$ and $F^0(t,u)$ within the equations in (1)–(7). The following list gives the order in which the functions can be eliminated, together with the equations that are solved for the specific function. $D(t,0)$, (3); $D(t,u)$, (2); $F(t,u)$, (6); $F^0(t,u)$, (7); $E(t,u)$, (4); $E^{+}(t,u)$, (5).

The modified equation (1) where all functions except for $C(t,u)$ and $C(t)$ are eliminated is

$$0 = \frac{r_1\,C(t,u)^2 + r_2\,C(t)^2 + r_3\,C(t,u)\,C(t) + r_4\,C(t,u) + r_5\,C(t) + r_6}{s_1\,C(t,u) + s_2\,C(t) + s_3}\text{, where} \tag{8}$$

$$r_1(t,u) := 2tu + 2t^2u + 2tu^2 + 2t^2u^2\,,$$
$$r_2(t,u) := 4t^2 + 4t^3 + 4t^2u + 4t^3u\,,$$
$$r_3(t,u) := -4t^2 - 4t^3 - 2tu - 6t^2u - 4t^3u - 2tu^2 - 2t^2u^2\,,$$
$$r_4(t,u) := 2t + 2t^2 + 4t^3 - u + tu + 4t^3u + u^2 + tu^2 - 2t^2u^2\,,$$
$$r_5(t,u) := -2t - 2t^2 - 4t^3 - 4tu - 2t^2u - 4t^3u + 2t^2u^2\,,$$
$$r_6(t,u) := u + 2tu + 2t^2u - tu^2\,,$$
$$s_1(t,u) := 2t^2u + 2t^2u^2\,,$$
$$s_2(t,u) := -2t^2 - 2t^3 + 2tu - 2t^2u - 2t^3u - 2t^2u^2\text{, and}$$
$$s_3(t,u) := t + t^2 + 2t^3 - u - tu - t^2u + 2t^3u + tu^2\,.$$

We now look for a solution $C(t,u)$ of (8), such that the numerator equals zero for all t and u and the denominator differs from zero. As $C(t,u)$ and $C(t)$ are both of degree two in (8), we can rewrite the equation as

$$(9)\qquad 0 = \big(\, g_1(t,u)\, C(t,u) + g_2(t,u) \,\big)^2 - g_3(t,u)\,, \qquad \text{where}$$

$$\begin{aligned}
g_1(t,u) &:= 4tu(t{+}1)(u{+}1)\,,\\
g_2(t,u) &:= 2t{+}2t^2{+}4t^3{-}u{+}tu{+}4t^3u{+}u^2{+}tu^2{-}2t^2u^2 - 2t(t{+}1)(u{+}1)(2t{+}u)\, C(t)\,,\\
g_3(t,u) &:= 4t^4(u{+}1)^2(4t^2{-}4tu{+}u^2{+}4t{-}4u{+}5){+}2tu(u^3{-}4u^2{-}3u{-}2){+}u^2(u{-}1)^2\\
&\quad {+}4t^3(u^4{-}5u^3{-}9u^2{-}u{+}2){+}t^2(5u^4{-}10u^3{-}15u^2{+}4)\\
&\quad + 4t^2(t{+}1)^2(u{+}1)^2(2t{-}u)^2\, C(t)^2 - 4t(t{+}1)(u{+}1)(4t^2{+}4t^3{+}8t^4\\
&\quad {-}4tu{-}4t^3u{+}8t^4u{-}u^2{-}5tu^2{-}2t^2u^2{-}8t^3u^2{+}u^3{+}tu^3{+}2t^2u^3)\, C(t)\,.
\end{aligned}$$

Both $C(t,u)$ and $C(t)$ appear in (9), and we cannot solve directly for one of these functions in t and u only. Setting $u = 0$ we only yield the trivial equation $0 = 0$. Instead, we apply the quadratic method due to Tutte [25], and follow the presentation in [15]. We assume that there exists a function $u_t := u(t)$ such that $g_3(t,u_t) = 0$. Equation (9) directly yields $0 = g_3(t,u_t) = (g_1\, C + g_2)^2(t,u_t)$, hence $0 = (g_1\, C + g_2)(t,u_t)$ and then $(\frac{\partial}{\partial u} g_3)(t,u_t) = \frac{\partial}{\partial u}(g_1 C + g_2)^2(t,u_t) = \big(2(g_1 C + g_2)\frac{\partial}{\partial u}(g_1 C + g_2)\big)(t,u_t) = 0$ holds as well. We now have the following pair of simultaneous equations: $0 = g_3(t,u_t)$ and $0 = (\frac{\partial}{\partial u} g_3)(t,u_t)$, depending on $C(t)$, t and u. We eliminate u by calculating the resultant, i.e., the Sylvester determinant, of $g_3(t,u_t)$ and $(\frac{\partial}{\partial u} g_3)(t,u_t)$ with respect to u, and obtain one polynomial in $C := C(t)$ and t, the roots of which include the common roots of $g_3(t,u_t)$ and $(\frac{\partial}{\partial u} g_3)(t,u_t)$; see [12] for details on resultants and generating functions. The resultant has several nontrivial factors, but only the following factor $p(C,t)$ will be relevant for us, as the other factors do not match the initial terms of $c(n)$.

$$\begin{aligned}
p(C,t) = {} & (8t^3{+}72t^4{+}264t^5{+}504t^6{+}528t^7{+}288t^8{+}64t^9)\, C^4\\
& + (12t^2{-}228t^3{-}988t^4{-}1756t^5{-}2032t^6{-}1792t^7{-}1024t^8{-}256t^9)\, C^3\\
& + (6t{+}218t^2{+}894t^3{+}2190t^4{+}3284t^5{+}3120t^6{+}2304t^7{+}1344t^8{+}384t^9)\, C^2\\
& + (1{-}43t{-}337t^2{-}1021t^3{-}1828t^4{-}2404t^5{-}2128t^6{-}1344t^7{-}768t^8{-}256t^9)\, C\\
& + ({-}1{+}36t{+}131t^2{+}350t^3{+}540t^4{+}616t^5{+}536t^6{+}304t^7{+}160t^8{+}64t^9)\,.
\end{aligned}$$

As the order of $p(C,t)$ as a polynomial in C is four, and $p(C,t) = 0$ yields four algebraic solutions for C. Comparing initial coefficients, we find that the following is the explicit form of the generating function $C(t)$.

$$
\begin{aligned}
C(t) &= \frac{1}{2q_1(t)}\Big(q_2(t) + \sqrt{h(t)} + \big(3q_3(t) - h(t) + \frac{q_4(t)}{\sqrt{h(t)}}\big)^{\frac{1}{2}}\Big) \\
h(t) &= q_1(t)\big((-1)^{\frac{1}{3}}(q_5(t) - q_6(t))^{\frac{1}{3}} - (q_5(t) + q_6(t))^{\frac{1}{3}}\big)\Big(\frac{2}{t}\Big)^{\frac{2}{3}} + q_3(t) \\
q_1(t) &= 12\,t(1+t)(1+2t)^3 \\
q_2(t) &= 3\,(-3 + 63t + 124t^2 + 128t^3 + 128t^4 + 64t^5) \\
q_3(t) &= 3\,(3 - 2126t + 1571t^2 + 11800t^3 + 9392t^4 + 256t^5 - 1024t^6) \\
q_4(t) &= 54\,(1 + 2681t - 46609t^2 - 96397t^3 + 48468t^4 + 188304t^5 \\
&\qquad + 62016t^6 - 63488t^7 - 32768t^8) \\
q_5(t) &= -729 - 49113t - 61936t^2 - 137856t^3 + 6144t^4 + 8192t^5 \\
q_6(t) &= (t-1)\big(-\frac{3}{2}(32t + 17 - 7\sqrt{7})(32t + 17 + 7\sqrt{7})\big)^{\frac{3}{2}}.
\end{aligned}
$$

An explicit form for $C(t,u)$ can be obtained by solving equation (9) for $C(t,u)$, and substituting $C(t)$ by its explicit form.

Having the algebraic equation at hand, we can apply singularity analysis: The dominant singularity lies in the exceptional set of the algebraic curve, and can be computed by evaluating the resultant R of $p(C,t)$ and $\frac{\partial}{\partial C}p(C,t)$ with respect to C. The solutions for t in the equation $R = 0$ can be computed symbolically with Mathematica, and the smallest real solution t_0, where additionally the equations $p(C,t_0) = 0$ and $\frac{\partial}{\partial C}p(C,t_0) = 0$ have a simultaneous solution, is a dominant singularity of $C(t)$. In this way, it is easy to compute the dominant singularity of $C(t)$, which is at $t_0 = 1/32(7\sqrt{7} - 17) \doteq 0.047508$ (that was computed before from the equations of Mullin and Schellenberg; see [2]), and proves the following.

Theorem 1 (essentially from [2])**.** *The number of c-nets $c(n)$ is in $(1/t_0)^{n+o(n)}$, where $1/t_0 = 16/27(17 + 7\sqrt{7}) \doteq 21.049042$.*

Using the Maple package GFUN [21], the algebraic equation $p(C,t)$ can be transformed automatically into a linear differential equation with polynomial coefficients, which in turn translates to a one parameter recurrence formula for c_n. Using Horner's method and this formula we computed the value of $c(100000)$ in 100 seconds on a PC.

Theorem 2. *For the coefficients $c(n)$ of $C(t)$ the following recursion holds.*

$$
\begin{aligned}
c(0) &= 1,\ c(1) = 1,\ c(2) = 7,\ c(3) = 73,\ c(4) = 879,\ c(5) = 11713,\\
&\quad c(6) = 167423,\ c(7) = 2519937,\ \textit{and for } n \geq 8\,,\\
c(n) &= \Big((42147840 + 49975296(n{-}7) + 19267584(n{-}7)^2 + 2408448(n{-}7)^3)\ c(n{-}7)\\
&\quad + (291529728 + 269461504(n{-}7) + 83615232(n{-}7)^2 + 8692736(n{-}7)^3)\ c(n{-}6)\\
&\quad + (533308032 + 435701440(n{-}7) + 119431200(n{-}7)^2 + 11026784(n{-}7)^3)\ c(n{-}5)\\
&\quad + (259749888 + 220560168(n{-}7) + 59988636(n{-}7)^2 + 5361276(n{-}7)^3)\ c(n{-}4)\\
&\quad + (-45552288 - 9821452(n{-}7) + 1941468(n{-}7)^2 + 418816(n{-}7)^3)\ c(n{-}3)\\
&\quad + (-16057320 - 11696062(n{-}7) - 2582841(n{-}7)^2 - 180467(n{-}7)^3)\ c(n{-}2)\\
&\quad + (5063688 + 2370408(n{-}7) + 367734(n{-}7)^2 + 18930(n{-}7)^3)\ c(n{-}1)\Big)\\
&\quad / \ (255024 + 99918(n{-}7) + 13041(n{-}7)^2 + 567(n{-}7)^3)\,.
\end{aligned}
$$

5. SAMPLING

We now discuss how to use the presented decomposition to sample c-nets uniformly at random. (As usual, $\tilde{O}(\cdot)$ denotes growth up to logarithmic factors.) Note that the analysis of [13] applies to expected running time, whereas our bound is deterministic. Moreover, they have parameters for vertices and faces, whereas we have parameters for the number of vertices and the size of the outer face. Thus the results are not directly comparable. Their upper bound is $O(n^4)$ for n vertices, and reduces to $O(n)$ if the ratio of vertex number to face number is fixed to a constant. The worst case is attained for triangulations.

Theorem 3. *There exists a deterministic polynomial time algorithm to sample c-nets on a given number of vertices and a given number of vertices on the outer face uniformly at random. The algorithm runs in $\tilde{O}(n^5)$ time and $O(n^3)$ space. If we allow a pre-computation, the algorithm can sample a c-net in $\tilde{O}(n^2)$ time and $O(n^5)$ space.*

Proof. The decomposition yields recursive counting functions for c-nets, d-nets, e-nets, e^+-nets, f-nets, and f^0-nets. For all $n, k \geq 0$:

$$
\begin{aligned}
c(n,k) &= \begin{cases} 1 & \text{if } n = k = 0\,, \\ d(n,k) + e(n,k) + f(n,k) & \text{else.} \end{cases} \\
d(n,k) &= c(n-1,k+1) + d(n-1,k+1)\,. \\
e(n,k) &= e^{+}(n,k) + d(n,k-1) + f^{0}(n-1,k) + f(n,k-1)\,. \\
e^{+}(n,k) &= e(n-1,k-1) + e(n-1,k) + c(n,k-1) + f^{0}(n-1,k)\,. \\
f(n,k) &= 2\sum_{i=0}^{n}\sum_{j=0}^{k} d(i,j)e^{+}(n-i,k-j)\,. \\
f^{0}(n,k) &= 2\sum_{i=0}^{n} d(i,0)e^{+}(n-i,k)\,.
\end{aligned}
$$

By induction on the lexicographically ordered pair (n,k), the decomposition reduces to the initial case within $O(nk)$ steps of recursion. Hence we can evaluate the functions using dynamic programming. The representation size of all computed numbers is linear, because it is bounded by the logarithm of the number of unlabeled c-nets, which is $O(2^{O(n)})$ according to Theorem 1. Note that the functions d, e, e^{+}, f, and f^{0} are at most as large as c according to their definitions. Since we employ a constant number of two-dimensional tables, the total space requirement is $O(n^3)$. Concerning the running time, each summation runs over at most two indices, and for each summand we have to perform one multiplication with $O(n)$ bit numbers. We assume an $O(n \log n \log\log n)$ multiplication algorithm (see e.g., [7]). Thus the running time for the computation of the values is within $\tilde{O}(n^5)$.

The values in the dynamic programming tables can be used to make the correct probabilistic decisions in a recursive construction of c-nets, which is essentially the inversion of the presented decomposition – this method is standard and known as the *recursive method* for sampling [8, 11, 20]. For each entry, we scan over all the entries from which it was computed (there are at most n^2 of them). We compute partial sums in another pass over these entries and build a balanced binary tree, where in each internal node the maximum over its left-hand siblings is stored. This will take $O(n^5)$ time in total, since we have $O(n^2)$ table entries, each with $O(n^2)$ dependencies, and each tree node stores an $O(n)$ bit number. After that, when given a random number between 1 and the maximum (i. e., the value of the entry for which the tree was built), we can find the corresponding table entry in one pass through the tree, while reading each bit of the random number only a constant number of times, and hence in $O(n)$ time. Then the procedure calls itself recursively. In the case of f and f^{0}, we have to trace back two separate lines, as the random sibling corresponds to a choice of the summation indices i (for f), respectively (i,j) (for f^{0}) and the actual summand is a product of two entries (e.g., $d(i,j)$ and $e^{+}(n-i,k-j)$ for $f(n,k)$ and (i,j)). Note that the sum of the bit lengths of both factors is linear in the bit length of the entry. It follows that the total running time for generating the decomposition tree is $\tilde{O}(n^2)$. If the decomposition tree is stored appropriately, we can output the sampled random graph in $O(n)$ time.

It is not necessary to create the binary trees physically for each entry of the tables. Instead, we can just redo the computations from the preprocessing and stop if the partial sum exceeds the random number. This way, the algorithm uses $\tilde{O}(n^5)$ time and $O(n^3)$ space. □

To sample unlabeled, unrooted 3-connected planar graphs uniformly at random, we apply *rejection sampling*. That is, we generate a c-net uniformly at random, but the resulting graph is accepted only with a probability that is inverse proportional to the size of the orbit of the root edge together with an incident face in the automorphism group of the graph. (It is well-known that the automorphism group of a planar graph can be computed efficiently, see e.g., [16].) If we do not output the graph, we restart the algorithm. Clearly, the output of this procedure are uniform random samples from the class of all 3-connected planar graphs. Since a 3-connected planar graph has with high probability a trivial automorphism group [3], the expected number of restarts is constant.

Corollary 1. *Using rejection sampling, we can sample 3-connected planar graphs using the algorithm of Theorem 3 in an expected constant number of rounds.*

6. Conclusion

Our main structural result is a new decomposition of rooted 3-connected planar graphs, which can easily be expressed in terms of recursive counting formulas, or equations for their generating functions. We use these equations to derive an algebraic equation of degree four that determines the generating function for the number of rooted 3-connected planar graphs on n vertices. Here we apply computer algebra systems, and also derive a single parameter recurrence formula, which allows to compute these numbers for much larger n than the previously known formulas of Mullin and Schellenberg [19].

The main algorithmic result is the first deterministic polynomial time algorithm to sample c-nets with a given number of vertices and a given size of the outer face uniformly at random. Since the recurrences of the decomposition do not involve any subtractions, the decomposition immediately translates into a sampling algorithm that produces a rooted 3-connected planar graph uniformly at random. The recursive counting formulas were implemented by top-down dynamic programming in C++ using the GMP library for exact arithmetic [14]. A table for small values of n and k is given in Figure 7.

It is fairly straightforward to see that the decomposition can be refined to control more parameters of the graph, e. g., the number of edges, or the degree of a root vertex [17]. Each parameter comes at the cost of another dimension in the tables and hence increases the pre-computation time by a quadratic factor. The recursive counting formulas with an additional parameter for the number for edges were also implemented, and we used the numbers of Mullin and Schellenberg [19] to check both implementations.

The algorithm can be used to obtain a faster and now fully deterministic polynomial time sampler for labeled planar graphs [4]. Also, using the rejection sampling method, we obtain an expected polynomial time algorithm for 3-connected planar graphs (isomorphism types of convex polyhedra).

$c(n,k)$	0	1	2	3	4	5	$n=6$
0	1	1	7	73	879	11713	167423
1	1	6	56	640	8256	115456	1710592
2	1	16	208	2848	41216	624384	9812992
3	1	30	560	9440	156592	2613664	44169600
4	1	48	1240	25864	496944	9234368	169378560
5	1	70	2408	61712	1377600	28663040	574139904
6	1	96	4256	132480	3430528	80104448	1758695424
7	1	126	7008	261648	7826544	205083936	4944057984
8	1	160	10920	483080	16600944	487362496	12906193920
9	1	198	16280	843744	33111232	1086226944	31579350528
$k=10$	1	240	23408	1406752	62659200	2289692416	72985375744

$c(n,k)$	7	8	9	$n=10$
0	2519937	39458047	637446145	10561615871
1	26468352	423641088	6966960128	117148778496
2	158883840	2636197888	44640468992	769058340864
3	756712960	13136471040	230851792896	4102116843520
4	3095526912	56624998400	1039080697856	19147850612736
5	11259283200	218198045184	4201424145408	80643838062592
6	37158281984	765948707328	15534537453568	311681600004096
7	112834665216	2481031718144	53154302311936	1117907385569280
8	318621198720	7487670554880	169818439763968	3751908804540416
9	843790483712	21217661003264	510172604564480	11860405982539776
$k=10$	2110406347008	56815355557376	1449735177678848	35506327812194304

FIGURE 7. A table of $c(n,k)$ for small c-nets on up to 23 vertices. The number of vertices on the outer face is $k+2$. The total number of vertices is $n+k+3$.

REFERENCES

[1] C. Banderier, P. Flajolet, G. Schaeffer, and M. Soria, *Random maps, coalescing saddles, singularity analysis, and Airy phenomena*, Random Struct. Algorithms **19** (2001), 194–246.

[2] E.A. Bender and L.B. Richmond, *The asymptotic enumeration of rooted convex polyhedra*, J. Combin. Theory Ser. B **36** (1984), 276–283.

[3] E.A. Bender and N.C. Wormald, *Almost all convex polyhedra are asymmetric*, Can. J. Math. **27** (1985), no. 5, 854–871.

[4] M. Bodirsky, C. Gröpl, and M. Kang, *Generating labeled planar graphs uniformly at random*, Proceedings of the 30th International Colloquium on Automata, Languages and Programming (ICALP'03), 2003, pp. 1095–1107.

[5] ______, *Sampling unlabeled biconnected planar graphs*, Proceedings of the 16th Annual International Symposium on Algorithms and Computation (ISAAC'05), Springer Verlag, 2005, pp. 593–603.

[6] M. Bousquet-Mélou, *On (some) functional equations arising in enumerative combinatorics*, extended abstract for FPSAC'01, 2001.

[7] P. Bürgisser, M. Clausen, and M.A. Shokrollahi, *Algebraic complexity theory*, Grundlehren der mathematischen Wissenschaften, no. 315, Springer Verlag, 1997.

[8] A. Denise and P. Zimmermann, *Uniform random generation of decomposable structures using floating-point arithmetic*, Theoretical Computer Science **218** (1999), 233–248.

[9] R. Diestel, *Graph theory*, Springer–Verlag, New York, 1997.

[10] P.J. Federico, *The number of polyhedra*, Philips Res. Repts **30** (1975), 220–231.

[11] P. Flajolet, P. Zimmerman, and B. Van Cutsem, *A calculus for the random generation of labelled combinatorial structures*, Theoretical Computer Science **132** (1994), no. 1-2, 1–35.

[12] P. Flajolet and R. Sedgewick, *Analytic combinatorics — symbolic combinatorics*, book in preparation (Oct. 2006), and Rapport de recherche 4103, INRIA, 2001.

[13] E. Fusy, D. Poulalhon, and G. Schaeffer, *Dissections and trees: applications to optimal mesh encoding and random sampling*, Proceedings of the sixteenth annual ACM–SIAM symposium on Discrete algorithms (SODA'05), Society for Industrial and Applied Mathematics, 2005, pp. 690–699.

[14] *The GNU multiple precision arithmetic library, version 4.1.2*, `http://swox.com/gmp/`.

[15] I.P. Goulden and D.M. Jackson, *Combinatorial enumeration*, John Wiley, New York, 1983.

[16] J.E. Hopcroft and R.E. Tarjan, *A v^2 algorithm for determining isomorphism of planar graphs*, Information Processing Letters **1** (1971), no. 1, 32–34.

[17] D. Johannsen, *Sampling rooted 3-connected planar graphs in deterministic polynomial time*, Diplomarbeit, Humboldt–Universität zu Berlin, 2006.

[18] V.A. Liskovets and T.R. Walsh, *Counting non-isomorphic 2-connected planar maps*, Graph Theory Newsletter **9** (1980), no. 6, 3.

[19] R.C. Mullin and P.J. Schellenberg, *The enumeration of c-nets via quadrangulations*, J. Combin. Theory **4** (1968), 259–276.

[20] A. Nijenhuis and H.S. Wilf, *Combinatorial algorithms*, Academic Press Inc., 1979.

[21] B. Salvy and P. Zimmermann, *GFUN: a Maple package for the manipulation of generating and holonomic functions in one variable*, ACM Trans. Math. Software **20** (1994), no. 2, 163–177.

[22] G. Schaeffer, *Conjugaison d'arbres et cartes combinatoires aléatoires*, Ph.D. thesis, Université Bordeaux I, 1998.

[23] G. Schaeffer, *Random sampling of large planar maps and convex polyhedra*, Proceedings of the thirty-first annual ACM symposium on theory of computing (STOC'99) (Atlanta, Georgia), ACM Press, May 1999, pp. 760–769.

[24] E. Steinitz, *Polyeder und Raumeinteilungen*, Encyclopädie der mathematischen Wissenschaften **3** (1922), no. 9, 1–139.

[25] W.T. Tutte, *A census of planar triangulations*, Canad. J. Math. **14** (1962), 21–38.

[26] ———, *A census of planar maps*, Canad. J. Math. **15** (1963), 249–271.

[27] ———, *Graph theory*, Cambridge University Press, 1984.

[28] T. Walsh and V.A. Liskovets, *Ten steps to counting planar graphs*, Proceedings of the Eighteenth Southeastern International Conference on Combinatoris, Graph Theory, and Computing, Congr. Numer. **60** (1987), 269–277.

[29] T. Walsh, *Counting non-isomorphic three-connected planar maps*, J. Combin. Theory Ser. B **32** (1982), 33–44.

[30] N.C. Wormald, *On the number of planar maps*, Canad. J. Math. **33** (1981), no. 1, 1–11.

MANUEL BODIRSKY, HUMBOLDT-UNIVERSITÄT ZU BERLIN, GERMANY
E-mail address: `bodirsky@informatik.hu-berlin.de`

CLEMENS GRÖPL, FREIE UNIVERSITÄT BERLIN, GERMANY
E-mail address: `groepl@inf.fu-berlin.de`

DANIEL JOHANNSEN, HUMBOLDT-UNIVERSITÄT ZU BERLIN, GERMANY
E-mail address: `johannse@informatik.hu-berlin.de`

MIHYUN KANG, HUMBOLDT-UNIVERSITÄT ZU BERLIN, GERMANY
E-mail address: `kang@informatik.hu-berlin.de`