Efficient Generation of Graphical Partitions

Tiffany M. Barnes *
tiff@bvcd.csc.ncsu.edu

Carla D. Savage †
savage@cayley.csc.ncsu.edu

Department of Computer Science
Box 8206
North Carolina State University
Raleigh, North Carolina 27695-8206

Abstract

Given a positive even integer n, we show how to generate the set $G(n)$ of graphical partitions of n, that is, those partitions of n which correspond to the degree sequences of simple, undirected graphs. The algorithm is based on a recurrence for $G(n)$, and the total time used by the algorithm, independent of output, is $O(|G(n)|)$, which is constant average time per graphical partition. This is the first algorithm shown to achieve such efficiency for generating $G(n)$ and the direct approach differs from earlier "generate and reject" schemes and the "interval/gap" approach.

1 Introduction

A *partition* π of a positive integer n is a sequence $\pi = (\pi_1, \pi_2, \ldots, \pi_l)$ of positive integers satisfying $\pi_1 \geq \pi_2 \geq \ldots \geq \pi_l$ and $\pi_1 + \pi_2 + \ldots + \pi_l = n$. A partition π of an even integer n is called *graphical* if it is the degree sequence of some simple undirected graph. For example, $\pi = (5,4,4,3,3,1)$ is graphical, whereas $\pi = (5,4,4,2,2,1)$ is not. We let $G(n)$ denote the set of graphical partitions of n. For convenience, we will call the empty partition, λ, graphical, so that $|G(0)| = 1$.

Earlier work has been concerned both with counting $G(n)$ and with generating the elements of $G(n)$. Most of this work is based on some necessary and sufficient condition

*Research supported by the National Science Foundation through the CRA Distributed Mentor Project, 1994

†Research supported in part by National Science Foundation Grant No. DMS9302505 and DIMACS

for a partition to be graphical. Several of these conditions, including the one below, are surveyed and shown to be equivalent in [SH].

Theorem 1 (Erdös and Gallai [EG]) *A positive integer sequence* $(\pi_1, \pi_2, \ldots, \pi_l)$*, with* $\pi_1 \geq \pi_2 \geq \ldots \geq \pi_l$*, is graphical if and only if* $\pi_1 + \pi_2 + \ldots + \pi_l$ *is even and for* $1 \leq j \leq l$*,*

$$\sum_{i=1}^{j} \pi_i \;\leq\; j(j-1) + \sum_{i=j+1}^{l} \min\{j, \pi_i\}.$$

To either count or generate $G(n)$, one could generate all partitions of n and use Theorem 1 to test each and delete those which are not graphical. It is possible to implement this test in time $O(n)$, giving a total time of $O(n|P(n)|)$, where $P(n)$ is the set of all partitions of n. Thus the time spent per element of $G(n)$ is $O(n|P(n)|/|G(n)|)$. How efficient this is depends on two things. First, the $O(n)$ time spent per test may be a gross overestimate, so that a more careful amortized analysis would perhaps show that on the average, the time spent testing an element of $P(n)$ was constant. Secondly, the efficiency of this generating scheme depends on the asymptotic behavior of the ratio $|P(n)|/|G(n)|$. However, to determine the limiting behavior of $|G(n)|/|P(n)|$ is an open problem, originally posed by H. Wilf. Some upper and lower bounds are known: A result of Erdös and Richmond [ER] establishes that

$$\overline{\lim}_{n\to\infty} \frac{\sqrt{n}\,|G(n)|}{|P(n)|} \;\geq\; \frac{\Pi}{\sqrt{6}}$$

(where Π here denotes the area of the unit circle) and Rousseau and Ali [RA] show that

$$\overline{\lim}_{n\to\infty} \frac{|G(n)|}{|P(n)|} \;\leq\; .25.$$

If the elements of $P(n)$ are generated in lexicographic order, it has been observed that the graphical partitions appear in groups of contiguous subsequences, called *intervals*, separated by *gaps* of non-graphical partitions. Earlier work attempted to identify partitions at the boundaries of these subsequences in order to either "jump" the gaps or to count the partitions in the intervals [MS]. In [St], the number of graphical partitions of even n with exactly l parts is tabulated for $1 \leq l \leq n/2 \leq 27$ and the largest value appearing in the table is 12,287, when $l = 16$ and $n = 54$.

Both counting and generating can now be done without explicit consideration of intervals and gaps. In [BS] we developed a recurrence which allowed $|G(n)|$ to be computed in time

$O(n^4)$ using $O(n^3)$ space. This was a substantial improvement over both the "generate and test" approach, which requires time $\Omega(|P(n)|)$, and the approach of identifying/counting partitions in intervals, since the number of intervals appears to grow faster than polynomial. In [BS], $|G(n)|$ is tabulated for even n through $n = 220$ and $|G(220)| = 7,443,670,977,177$. However, the recurrence of [BS] is only for counting, and not for generating, $G(n)$.

In this paper, we use a different recurrence, for the *set* $G(n)$, to generate directly the elements of $G(n)$ in total time $O(|G(n)|)$, averaging constant time per element of $G(n)$, which is asymptotically optimal. This analysis does not count the time to explicitly print out the graphical partitions as they are computed. The recursive algorithm is developed in Section 2. In Section 3, we show how to implement the algorithm in constant average time per element. In the remainder of this section we discuss the criterion we will use to test whether a partition is graphical.

For a partition $\pi = (\pi_1, \ldots, \pi_l)$, the associated *Ferrers graph* is an array of l rows of dots, where row i has π_i dots and rows are left justified. Let π' denote the conjugate partition $\pi' = (\pi'_1, \ldots, \pi'_m)$ where $m = \pi_1$ and π'_i is the number of dots in the i-th column of the Ferrers graph of π. The *Durfee square* of π is the largest square subarray of dots in the Ferrers graph of π. Let $d(\pi)$ denote the size (number of rows) of the Durfee square of π. The sequence

$$(\pi_1 - \pi'_1, \quad \pi_2 - \pi'_2, \ \ldots, \ \pi_{d(\pi)} - \pi'_{d(\pi)})$$

is the sequence of *successive ranks* of π [At]. It will be convenient to work with the negatives of the ranks, so, for $1 \leq i \leq d(\pi)$, let $r_i(\pi) = \pi'_i - \pi_i$. We call $\mathbf{r}(\pi) = [r_1(\pi), \ldots r_{d(\pi)}(\pi)]$ the *corank* vector of π.

The necessary and sufficient condition below, attributed to Nash-Williams, is proved in [RA] and [SH].

Theorem 2 [Nash-Williams] *A partition π of an even integer is graphical if and only if for $1 \leq j \leq d(\pi)$,*

$$\sum_{i=1}^{j} r_i(\pi) \geq j.$$

It can be shown that for $1 \leq j \leq d(\pi)$, the j-th Nash-Williams condition is equivalent to the j-th Erdös-Gallai condition. Furthermore, if conditions $1, 2, \ldots, d(\pi)$ of Erdös-Gallai are satisfied, then so are the remaining Erdös-Gallai conditions [RA].

2 A Recurrence for $G(n)$

We begin with some notation. For a partition $\pi = (\pi_1, \pi_2, \ldots, \pi_l)$, define the *size* of π, denoted $|\pi|$, to be the sum of its parts: $|\pi| = \pi_1 + \pi_2 + \ldots + \pi_l$. Define head$(\pi) = \pi_1$ and tail$(\pi) = \pi_l$, to be the first and last parts of π, respectively. If $\beta = (\beta_1, \beta_2, \ldots, \beta_s)$ is another partition and if tail$(\pi) \geq$ head(β), then $\pi \cdot \beta$ is the partition $(\pi_1, \pi_2, \ldots, \pi_l, \beta_1, \beta_2, \ldots, \beta_s)$ of the integer $|\pi|+|\beta|$. If tail$(\pi) \geq i$, then we use $\pi \cdot (i)$ to denote the partition $(\pi_1, \pi_2, \ldots, \pi_l, i)$. For $k \geq 1$, $\pi \cdot 1^k$ is the partition $\pi \cdot (1) \cdot (1) \cdot \ldots \cdot (1)$ with k ones appended to the end of π.

For $n \geq k \geq 1$, let $P(n,k)$ be the set of all partitions π of n with head$(\pi) \leq k$. Extend $P(n,k)$ to all integers n, k by: $P(n,k) = \emptyset$ if $n < 0$ or $k < 0$ or if both $n = 0$ and $k < 0$; otherwise, $P(0,k) = \{\lambda\}$. (Recall that λ denotes the empty partition.) Furthermore, $P(n,k) = P(n,n)$ if $k > n$.

For a partition, α, and for $n \geq k \geq 1$, define $G(\alpha, n, k)$ to be the set of all graphical partitions of $|\alpha| + n$ which have the form $\alpha \cdot \beta$ for some $\beta \in P(n,k)$ with head$(\beta) \leq$ tail(α). As was done for $P(n,k)$, extend $G(\alpha, n, k)$ to all integers n, k by: $G(\alpha, n, k) = \emptyset$ if $n < 0$ or $k < 0$ or if both $n = 0$ and $k < 0$; otherwise, $G(\alpha, 0, k) = \{\alpha\}$ if α is graphical and $G(\alpha, 0, k) = \emptyset$ if α is not graphical. If $k > n$, then $G(\alpha, n, k) = G(\alpha, n, n)$.

Note that if tail$(\alpha) \geq k$ and if $n \geq k \geq 1$, then any $\pi \in G(n, \alpha, k)$ has the form $\pi = \alpha \cdot \beta$ for some $\beta \in P(n,k)$. Since the first part of β, namely β_1, satisfies $\beta_1 = i$ for some i such that $1 \leq i \leq k$, π can be written as $\pi = \alpha \cdot (i) \cdot \delta$, where $\beta = (i) \cdot \delta$ and $\delta \in P(n-i, i)$. Thus, $G(\alpha, n, k)$ can be written as the disjoint union

$$G(\alpha, n, k) = \bigcup_{i=1}^{k} G(\alpha \cdot (i), n-i, i). \tag{1}$$

Furthermore, for n even, $G(n) = G(\lambda, n, n)$. (In fact, by Theorem 2, $G(n) = G(\lambda, n, n/2)$).

Our algorithm is based on the recurrence (1). We will successively consider each set,

$$G(\alpha \cdot (1), n-1, 1), \quad G(\alpha \cdot (2), n-2, 2), \ \ldots \tag{2}$$

and perform a test to determine if the set is empty. If nonempty, we generate the set recursively.

The test is based on the following lemma.

Lemma 1 *For a partition, α, and for $s \geq 1$, $t \geq 0$, the set $G(\alpha, t, s)$ is nonempty if and only if the partition $\alpha \cdot 1^t$ is graphical.*

Proof. If $t = 0$, $G(\alpha, 0, s)$ is empty if $\alpha = \alpha \cdot 1^0$ is not graphical and otherwise, it contains only $\alpha = \alpha \cdot 1^0$. For $t > 0$, if $\alpha = \lambda$, then by Theorem 2, $G(\lambda, t, s)$ contains 1^t as long as $t \geq 2$, otherwise 1^t is not graphical and $G(\lambda, s, t)$ is empty.

Assume $t > 0$ and $\alpha \neq \lambda$. Clearly, if $\alpha \cdot 1^t$ is graphical, then it is a member of $G(\alpha, t, s)$, which is therefore nonempty. For the converse, suppose that $\pi \in G(\alpha, t, s)$. Then by Theorem 2, for $1 \leq j \leq d(\pi)$,

$$\sum_{i=1}^{j} r_i(\pi) \geq j. \tag{3}$$

Since π can be formed by adding t dots in some way to the Ferrers graph of α,

$$t + \sum_{i=1}^{j} r_i(\alpha) \geq \sum_{i=1}^{j} r_i(\pi) \tag{4}$$

for $1 \leq j \leq d(\alpha)$. (Note $d(\alpha) \leq d(\pi)$.) Since $\alpha \neq \lambda$, $d(\alpha) = d(\alpha \cdot 1^t)$ and for $1 \leq j \leq d(\alpha)$,

$$\sum_{i=1}^{j} r_i(\alpha \cdot 1^t) = t + \sum_{i=1}^{j} r_i(\alpha). \tag{5}$$

It follows then from (3) - (5) and Theorem 2 that $\alpha \cdot 1^t$ is graphical. ■

We show in Lemma 2 below that if one of the sets in (2) is found to be empty, then so are all succeeding sets.

Lemma 2 *For a partition, α, for $t \geq 1$, and for $1 \leq s \leq tail(\alpha) - 1$, if the partition $\pi = \alpha \cdot (s) \cdot 1^t$ is not graphical, then neither is the partition $\sigma = \alpha \cdot (s+1) \cdot 1^{t-1}$.*

Proof. Let $d(\alpha) = k$ and $\mathbf{r}(\alpha) = [r_1, r_2, \ldots, r_k]$. Then $d(\pi) \in \{k,\ k+1\}$ and $d(\sigma) \in \{d(\pi),\ k+1\}$. If $d(\pi) = k$ then $s \leq k$ and

$$\mathbf{r}(\pi) = [r_1 + 1 + t,\ r_2 + 1,\ \ldots,\ r_s + 1,\ r_{s+1},\ \ldots,\ r_k] \tag{6}$$

and the first k coranks in $\mathbf{r}(\sigma)$ are

$$[r_1 + t,\ r_2 + 1,\ \ldots,\ r_{s+1} + 1,\ r_{s+2},\ \ldots,\ r_k]. \tag{7}$$

Recall that α' is the conjugate of α. If $d(\pi) = k + 1$ then $\alpha'_{k+1} = k$, $\pi_{k+1} = s \geq k + 1$, $d(\sigma) = k + 1$, and $\sigma_{k+1} = s + 1$, so,

$$\mathbf{r}(\pi) = [r_1 + 1 + t, r_2 + 1, \ldots, r_k + 1, (k+1) - s], \tag{8}$$

$$\mathbf{r}(\sigma) = [r_1 + t, r_2 + 1, \ldots, r_k + 1, (k+1) - (s+1)]. \tag{9}$$

Since, by hypothesis, π is not graphical, by Theorem 2, there is a j^*, $1 \leq j^* \leq d(\pi)$ such that

$$\sum_{i=1}^{j^*} r_i(\pi) < j^*. \tag{10}$$

It can be checked from (6) - (10) that

$$\sum_{i=1}^{j^*} r_i(\sigma) \;=\; -c + \sum_{i=1}^{j^*} r_i(\pi) \;<\; -c + j^*$$

where

$$c = \begin{cases} 1 & \text{if } d(\pi) = k \text{ and } j^* \leq s \\ 0 & \text{if } d(\pi) = k \text{ and } j^* > s \\ 1 & \text{if } d(\pi) = k+1 \text{ and } j^* \leq k \\ 2 & \text{if } d(\pi) = k+1 \text{ and } j^* = k+1. \end{cases}$$

In all of these cases, $c \geq 0$, so,

$$\sum_{i=1}^{j^*} r_i(\sigma) \;<\; j^*,$$

and therefore, since $1 \leq j^* \leq d(\sigma)$, by Theorem 2, σ is not graphical. ■

The algorithm is summarized below. In Section 3, we describe and analyze an efficient implementation and, in particular, show how the test of line (**) can be done in only constant time.

GRAPHICAL-PARTITIONS(n)
{Main procedure: }
{Given nonnegative even integer, n, procedure generates all graphical partitions of n}
    if $n < 0$ then return($\emptyset$)
    else if $n = 0$ then return(λ)
    else GENERATE(λ, n, n)

procedure GENERATE(α, n, k)
{Assumes that $n \geq 0$, $k \geq 1$, tail(α) $\geq k$, and $G(\alpha, n, k) \neq \emptyset$.}
{These conditions hold when GENERATE is called by the main procedure and}

{they are guaranteed to hold in any recursive call to GENERATE.}

if $n = 0$ or $k = 1$ then output $(\alpha \cdot 1^n)$

else begin

if $k > n$ then $k \leftarrow n$;

$i \leftarrow 1$;

done $\leftarrow$ false;

while not done and $i \leq k$ do begin

{Determine whether $G(\alpha \cdot (i), n-i, i)$ is empty: }

if $\alpha \cdot (i) \cdot 1^{n-i}$ is not graphical (**)

then done $\leftarrow$ *true*

else GENERATE$(\alpha \cdot (i), n-i, i)$

$i \leftarrow i + 1$

end

end

3 Efficient Implementation

In this section, we first discuss how to implement the test of line (**) of the algorithm of Section 2 so that it requires only constant time. We then analyze the resulting algorithm to show that it requires total time $O(|G(n)|)$, ignoring the time for printing the output.

For a partition π, define $t(\pi)$ as follows. If π is graphical, then $t(\pi) = 0$. Otherwise, if π has corank vector $\mathbf{r}(\pi) = [r_1(\pi), \ldots, r_{d(\pi)}(\pi)]$, then $t(\pi)$ is the smallest integer s such that

$$s + \sum_{i=1}^{j} (r_i(\pi) - 1) \geq 0 \tag{11}$$

for all $1 \leq j \leq d(\pi)$. Now if, for $1 \leq j \leq d(\pi)$, $c_j(\pi)$ is defined by

$$c_j(\pi) = \sum_{i=1}^{j} (1 - r_i(\pi)), \tag{12}$$

then note that

$$t(\pi) = \max\{0,\ c_1(\pi), \ldots, c_{d(\pi)}(\pi)\}. \tag{13}$$

We call the vector $\mathbf{c}(\pi) = [c_1(\pi), \ldots, c_{d(\pi)}(\pi)]$ the *cumulative need* vector.

We make use of the quantity $t(\pi)$ in the following way. Note that by Theorem 2, the partition $\pi \cdot 1^x$ is graphical if and only if $x \geq t(\pi)$. Thus, for the test of line (**) in the algorithm of Section 2, we can determine whether $\alpha \cdot (i) \cdot 1^{n-i}$ is graphical by computing $t(\alpha \cdot (i))$ and then checking whether $n-i \geq t(\alpha \cdot (i))$. Theorem 3 below shows how to compute $t(\alpha \cdot (i))$ and some additional information about $\alpha \cdot (i)$ from the quantity $t(\alpha \cdot (i-1))$ and some additional information about $\alpha \cdot (i-1)$.

For a partition π and for $1 \leq j \leq d(\pi)$, define

$$m_j(\pi) = \max\{0,\ c_1(\pi), \ldots, c_j(\pi)\}. \tag{14}$$

Then

$$t(\pi) = m_{d(\pi)}(\pi). \tag{15}$$

Theorem 3 *Let α be a partition with $d = d(\alpha)$. Define $\alpha \cdot (0)$ to be α and let $r_0(\pi) = c_0(\pi) = m_0(\pi) = 0$ for any partition π. Then*

(i) For $1 \leq i \leq d$ and $i \leq$ tail(α),

$$\begin{aligned}
&r_i(\alpha \cdot (i)) = r_i(\alpha \cdot (i-1)) + 1;\\
&c_i(\alpha \cdot (i)) = c_{i-1}(\alpha \cdot (i-1)) + 1 - r_i(\alpha \cdot (i));\\
&m_i(\alpha \cdot (i)) = \max\{m_{i-1}(\alpha \cdot (i-1)), c_i(\alpha \cdot (i))\};\\
&t(\alpha \cdot (i)) = \max\{m_i(\alpha \cdot (i)), t(\alpha \cdot (i-1)) - 1\}.
\end{aligned}$$

(ii) For $d+1 \leq i \leq$ tail(α),

$$\begin{aligned}
&r_{d+1}(\alpha \cdot (i)) = 0;\\
&c_{d+1}(\alpha \cdot (i)) = c_d(\alpha \cdot (i-1)) + 1;\\
&m_{d+1}(\alpha \cdot (i)) = \max\{m_d(\alpha \cdot (i-1)), c_{d+1}(\alpha \cdot (i))\};\\
&t(\alpha \cdot (i)) = \max\{m_{d+1}(\alpha \cdot (i)), t(\alpha \cdot (i-1))\}.
\end{aligned}$$

(iii) For $d+2 \leq i \leq$ tail(α),

$$\begin{aligned}
&r_{d+1}(\alpha \cdot (i)) = r_{d+1}(\alpha \cdot (i-1)) - 1;\\
&c_{d+1}(\alpha \cdot (i)) = c_{d+1}(\alpha \cdot (i-1)) + 1;\\
&m_{d+1}(\alpha \cdot (i)) = \max\{m_{d+1}(\alpha \cdot (i-1)), c_{d+1}(\alpha \cdot (i))\};\\
&t(\alpha \cdot (i)) = \max\{m_{d+1}(\alpha \cdot (i)), t(\alpha \cdot (i-1))\}.
\end{aligned}$$

Proof. Let α have corank vector $\mathbf{r}(\alpha) = [r_1, \ldots, r_d]$ and cumulative need vector $\mathbf{c}(\alpha) = [c_1, \ldots, c_d]$. Then for $1 \leq i \leq d$ and tail(α) $\geq i$, the partition $\alpha \cdot (i)$ has the following corank

and cumulative need vectors:

$$\mathbf{r}(\alpha \cdot (i)) = [r_1 + 1, r_2 + 1, \ldots, r_i + 1, r_{i+1}, \ldots, r_d], \tag{16}$$

$$\mathbf{c}(\alpha \cdot (i)) = [c_1 - 1, c_2 - 2, \ldots, c_i - i, c_{i+1} - i, \ldots, c_d - i]. \tag{17}$$

From this and (6) - (10), it can be verified that the theorem follows.

On the other hand, for $d < i \leq \text{tail}(\alpha)$, the size of the Durfee square becomes $d(\alpha \cdot (i)) = d + 1$ and the corank and cumulative need vectors are

$$\mathbf{r}(\alpha \cdot (i)) = [r_1 + 1, r_2 + 1, \ldots, r_d + 1, d - i + 1], \tag{18}$$

$$\mathbf{c}(\alpha \cdot (i)) = [c_1 - 1, c_2 - 2, \ldots, c_d - d, c_d - 2d + i] \tag{19}$$

and from this and (6) - (10) the theorem follows. ■

To compute $t(\alpha \cdot (i))$, we use Theorem 3 as follows. Define the invariant properties $\mathcal{A}(i)$ and $\mathcal{B}(i)$ as below:

Invariant Properties $\mathcal{A}(i)$:
(i) $\mathbf{r}$ is the corank vector of $\alpha(i)$
(ii) $c = c_i(\alpha \cdot (i))$
(iii) $m = m_i(\alpha \cdot (i))$
(iv) $t = t(\alpha \cdot (i))$

Invariant Properties $\mathcal{B}(i)$
(i) $\mathbf{r}$ is the corank vector of $\alpha(i)$
(ii) $c = c_{d+1}(\alpha \cdot (i))$
(iii) $m = m_{d+1}(\alpha \cdot (i))$
(iv) $t = t(\alpha \cdot (i))$

In the procedure GENERATE(α, n, k), if initially $\mathbf{r}$ is the rank vector of α, and if $d = d(\alpha)$, $t = t(\alpha)$, and $c = m = 0$, then $\mathbf{r}, d, c, m$ and t satisfy properties $\mathcal{A}(0)$. This initialization is achieved by passing $\mathbf{r}, d$ and t to GENERATE from the calling procedure. In its turn, GENERATE must pass the required values of these parameters in any recursive calls it makes.

Assume that $1 \leq i \leq d$ and $i \leq \text{tail}(\alpha)$ and $\mathbf{r}, d, t, m$ and c satisfy properties $\mathcal{A}(i)$. Then after performing the following operations, by Theorem 3(i), properties $\mathcal{A}(i+1)$ are satisfied:

(a) $r_i \leftarrow r_i + 1$
(b) $c \leftarrow c + 1 - r_i$

(c) $m \leftarrow \max\{m, c\}$

(d) $t \leftarrow \max\{m, t-1\}$

Assume that $\mathbf{r}, d, t, m$ and c satisfy properties $\mathcal{A}(d)$. Then by Theorem 3(ii), after executing the following steps (1) - (6), $\mathbf{r}, d, t, m$ and c satisfy properties $\mathcal{B}(d+1)$:

(1) $d \leftarrow d+1$

(2) $r_d \leftarrow 1$

(3) $r_d \leftarrow r_d - 1$

(4) $c \leftarrow c+1$

(5) $m \leftarrow \max\{m, c\}$

(6) $t \leftarrow \max\{m, t\}$

By Theorem 3(iii), if properties $\mathcal{B}(i)$ hold for $i \geq d+1$, then after steps (3) - (6), $\mathcal{B}(i+1)$ holds.

Thus, for any i satisfying $1 \leq i \leq \text{tail}(\alpha)$, the quantity $t = t(\alpha \cdot (i))$ is correctly computed from $t = t(\alpha \cdot (i-1))$ either by steps (a) - (d), or (1) - (6), or (3) - (6) above, depending on the value of i, relative to d.

Note that steps (a) - (d), as well as steps (1) - (6), take only constant time. Thus, computation of $t(\alpha \cdot (i))$, as well as $\mathbf{r}, d, c$, and m, for $\alpha \cdot (i)$, can be computed in constant time from those same quantities for $\alpha \cdot (i-1)$.

We perform an amortized analysis of the total time required to generate the elements of $G(n)$. The standard way to do this, if the algorithm is recursive, is to do the accounting on the tree of recursive calls. We show first that the number of nodes in the tree is $O(|G(n)|)$. The time spent by various parts of the algorithm is then, for accounting purposes, charged in packets to the nodes of the tree in such a way that the total time required by the algorithm is at most the sum of the charges at each node. Finally, we argue that the amount of time charged to each node of the tree is bounded by a constant, from which it follows that the total time is $O(|G(n)|)$.

We analyze the total time required to generate $G(n)$ by viewing the tree of recursive calls, neglecting the time for the output step. The call made by the main procedure is the root node and a node has one child for each recursive call it makes. As long as the root is not a leaf, each leaf in the tree corresponds to a graphical partition and conversely.

Furthermore, if a node is an only child of its parent, it must correspond to a procedure call of the form $G(\alpha, t, 1)$, which is a leaf. Thus, the number of non-leaf nodes with fewer than two children is no more than the number of leaves. This implies that the total number of nodes in the tree is less than 3 times the number of leaves $= 3|G(n)|$.

We now charge the work done by the algorithm to the nodes in the tree as follows. Each iteration of a loop takes constant work and culminates either in a recursive call (in which case the work is charged to the recursive call) or with the variable *done* being assigned the value *true* (in which case the work is charged to the node itself.) If no iteration of the loop is executed, then the node itself is charged (it is a leaf.) Finally, if the **r** vector is made global and must be restored before returning to the calling procedure, the number of changes made to **r** during a procedure is one more than the number of recursive calls made by that procedure. Charge one change to the node itself and one change to each of its children. In this accounting scheme, no node is charged for more than a constant number of operations. Thus the total time is at most a constant times the number of nodes which is $O(|G(n)|)$.

A PASCAL implementation of the algorithm is available from the second author. An implementation has also been installed as part of Frank Ruskey's *Combinatorial Object Server* which can be accessed at http://sue.csc.uvic.ca/~cos.

We note that generating $G(100)$ takes about 4 minutes on a Sparcstation and $|G(100)|$ = 69,065,657.

Acknowledgement We are grateful to the referees for several helpful suggestions to improve the presentation of this paper.

References

[At] A. O. L. Atkin, "A note on ranks and conjugacy of partitions," *Quart. J. Math., Oxford Ser. (2)***17** (1966) 335-338.

[BS] T. M. Barnes and C. D. Savage, "A recurrence for counting graphical partitions," *Electronic Journal of Combinatorics* 1, R11 (1995).

[EG] P. Erdös and T. Gallai, "Graphs with given degree of vertices," *Mat. Lapok* **11** (1960) 264-274.

[ER] P. Erdös and L. B. Richmond, "On graphical partitions," *Combinatorica* **13** (1993) 57-63.

[MS] N. Metropolis and P. R. Stein, "The enumeration of graphical partitions," *Europ. J. Combinatorics* 1 (1980) 139-153.

[RA] C. C. Rousseau and F. Ali, "A note on graphical partitions," *Journal of Combinatorial Theory, Series B* 64 (1995) 314-318.

[SH] G. Sierksma and H. Hoogeveen, "Seven criteria for integer sequences being graphic," *Journal of Graph Theory* **15**, No. 2 (1991) 223-231.

[St] P. R. Stein, "On the number of graphical partitions", *Proceedings of the Ninth Southeastern Conference of Combinatorics, Graph Theory, and Computing,* Congressus Numerantium XXI (1978) 671-684.