Algorithmic Combinatorics based on Slicing Posets

Vijay K. Garg[*]

Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, TX 78712-1084, USA
and
Department of Computer Science & Engineering
Indian Institute of Technology,
Kanpur, India

Abstract. *We show that some recent results in slicing of a distributed computation can be applied to developing algorithms to solve problems in combinatorics. A combinatorial problem usually requires enumerating, counting or ascertaining existence of structures that satisfy a given property B. We cast the combinatorial problem as a distributed computation such that there is a bijection between combinatorial structures satisfying B and the global states that satisfy a property equivalent to B. We then apply results in slicing a computation with respect to a predicate to obtain a small representation of only those global states that satisfy B. The slicing results are based on a generalization of Birkhoff's Theorem of representation of finite distributive lattices. This gives us an efficient (polynomial time) algorithm to enumerate, count or detect structures that satisfy B when the total set of structures is large but the set of structures satisfying B is small. We illustrate our techniques by analyzing problems in integer partitions, set families, and set of permutations.*

1 Introduction

Consider the following combinatorial problems:
(Q1) Count the number of subsets of the set $[n]$ (the set $\{1 \dots n\}$) which have size m and do not contain any consecutive numbers.
(Q2) Enumerate all integer partitions less than $(\lambda_1, \lambda_2, \dots, \lambda_n)$ in which the first part is equal to the second part.
(Q3) Give the number of permutations of $[n]$ in which $i \leq j$ implies that the number of inversions of i is atmost the number of inversions of j.

Our goal in this paper is to show how such problems can be solved *mechanically* and *efficiently* for any fixed values of the parameters n and m. It is important to note that someone trained in combinatorics may be able to solve all

[*] supported in part by the NSF Grants ECS-9907213, CCR-9988225, Texas Education Board Grant ARP-320, an Engineering Foundation Fellowship, and an IBM grant. `garg@ece.utexas.edu`

of these problems efficiently. Our emphasis is on techniques that can be applied mechanically. On the other hand, for the fixed values of n and m, all the sets above are finite and therefore all the problems can be solved mechanically. Our emphasis is on *efficiency*. To be more precise, let L be a large set of combinatorial structures (for example, all subsets of $\{1 \ldots n\}$ of size m, all permutations of $[n]$ etc.) Each combinatorial problem requires enumerating, counting, or searching the subset of structures that satisfy a given property B. Call this set $L_B \subseteq L$. For example, in the problem (Q1), L is the set of all subsets of $[n]$ of size m and L_B is the set of all subsets that do not contain any consecutive numbers from $[n]$. For any fixed set of parameters m and n, the size of L is large but finite, enabling one to enumerate all possible structures and then to check each one of them for the property B. This approach results in an algorithm that requires time proportional to the set L which is exponential in n (or m). This paper proposes a technique that provides answers to some combinatorial problems in polynomial time and for others, such as those involving enumeration, in time proportional to the size of L_B (and not L). Our technique is applicable whenever B satisfies a property called *regularity* and we give several examples of regular B in this paper.

To explain our technique, we use the term *small* to mean polynomial in n and m, and *large* to mean exponential in n or m. Thus, the set L is large. We first build a *small* structure P such that all elements of P can be generated by L. Second, we compute a *slice* of P with respect to B, denoted by P_B, such that P_B generates all elements of L_B and when B is regular only those elements. P_B is a small structure and can be efficiently analyzed to answer questions about L_B. For regular B, one could simply enumerate all elements of L_B from P_B.

Our approach is based on some recent results on *slicing* a distributed computation with respect to a predicate B [GM01,MG01]. Informally, a slice of a computation with respect to a predicate B is a subcomputation with the least number of global states that contains all global states that satisfy B. Slicing, in turn, is based on Birkhoff's Theorem of representation of finite distributive lattices [DP90]. The small structure P is a directed graph representing a distributed computation on n processes. The set of all (consistent) global states of the computation is the large structure L.

Consider any predicate B defined on L, or equivalently, the a subset L_B of L. B is called regular if L_B is a sublattice of L. From Birkhoff's theorem we know that there exists a poset that generates L_B. We show that every sublattice of L can be generated by a poset that can be obtained by adding edges to the poset P. Note that when edges are added to the graph of a poset cycles may form. In this case we simply consider the poset of strongly connected components in the graph. We denote the small structure obtained after adding edges to P as P_B. Now P_B can be used to enumerate elements in L_B, or to analyze the number of elements in L_B. Many algorithms have been proposed to enumerate ideals of a poset; for example by Steiner[Ste86] and Squire[Squ95]. In distributed computing, the algorithms to explore the global state lattice address the identical problem (see [CM91,AV01,Gar02]). Determining the count of the elements in L_B

given P_B is #P-complete for general posets [PB83] but can be done efficiently for 2-dimensional posets[Ste84].

We apply these ideas to many traditional problems in combinatorics. Due to the lack of space, all proofs in this paper are omitted and the interested reader can consult the technical report available at the author's website.

2 Notation and Definitions

A pair (X, P) is called a partially ordered set or poset if X is a set and P is a reflexive, antisymmetric, and transitive binary relation on X. We simply write P as a poset when X is clear from the context. We say $e \leq f$ in P when $(e, f) \in P$ and that f covers e if $e < f$ and there is no g such that $e < g < f$. Let $e, f \in X$ with $e \neq f$. If either $e < f$ or $f < e$, e and f are *comparable*. On the other hand, if neither $e < f$ nor $f < e$, then e and f are *incomparable*. A poset (X, P) is called *chain* if every distinct pair of points from X is comparable in P. Similarly, a poset is an *antichain* if every distinct pair of points from X is incomparable in P.

(X, P) and (Y, Q) are isomorphic, if there exists a $1-1$ and onto map $F : X \longrightarrow Y$ such that $c \leq d$ in P if and only if $F(c) \leq F(d)$ in Q. A poset (X, Q) is an extension of (X, P) if for all $e, f \in X$, $e < f$ in P implies $e < f$ in Q. (X, Q) is a *linear extension* if it is an extension of (X, P) and is a chain.

A *lattice* is a poset L such that for all $x, y \in L$, the least upper bound of x and y exists, called the *join* of x and y (denoted by $x \sqcup y$); and the greatest lower bound of x and y exists, called the *meet* of x and y (denoted by $x \sqcap y$). A *sublattice* is a subset of L closed under join and meet. A sublattice for which there exists two elements c and d such that it includes all x which lie between c and d (i.e. $c \leq x \leq d$) is called an *interval lattice* and denoted by $[c, d]$. A lattice L is *distributive* if for all $x, y, z \in X$: $x \sqcap (y \sqcup z) = (x \sqcap y) \sqcup (x \sqcap z)$.

Let (X, P) be a poset and let $G \subseteq X$. G is called an *order ideal* in (X, P) if $e \in G$ whenever $f \in G$ and $e \leq f$ in P. Consider the poset in Fig. 1(a). The set $\{b, d\}$ is an order ideal. The set $\{a, c\}$ is not because it includes c but does not include b which is smaller than c. We simply use *ideal* for order ideal in this paper. Let L denote the family of all ideals of P. Define a partial order on L by $G \leq H$ in L if and only if $G \subseteq H$. It is well known that the set of ideals forms a distributive lattice and conversely every finite distributive lattice can be constructed in this manner. Fig. 1(a) shows a poset and Fig. 1(b) its lattice of ideals. Given a finite distributive lattice L, one can determine the poset that generates L as follows. An element $e \in L$ is *join-irreducible* if it cannot be written as joins of two elements different from e (i.e., there is exactly one edge coming into the element, see Fig. 1(b)). For any $e \in P$, let $J(e)$ denote the least ideal in L that contains e. It is easy to show that $J(e)$ is join-irreducible. Let $J(L)$ denote the set of all join-irreducible elements in L. Birkhoff's theorem states that any finite distributive lattice L is isomorphic to the set of ideals of the poset $J(L)$ (and dually, any finite poset P is isomorphic to join-irreducible elements of the set of ideals of P). Meet-irreducible elements of L can be defined in an analogous

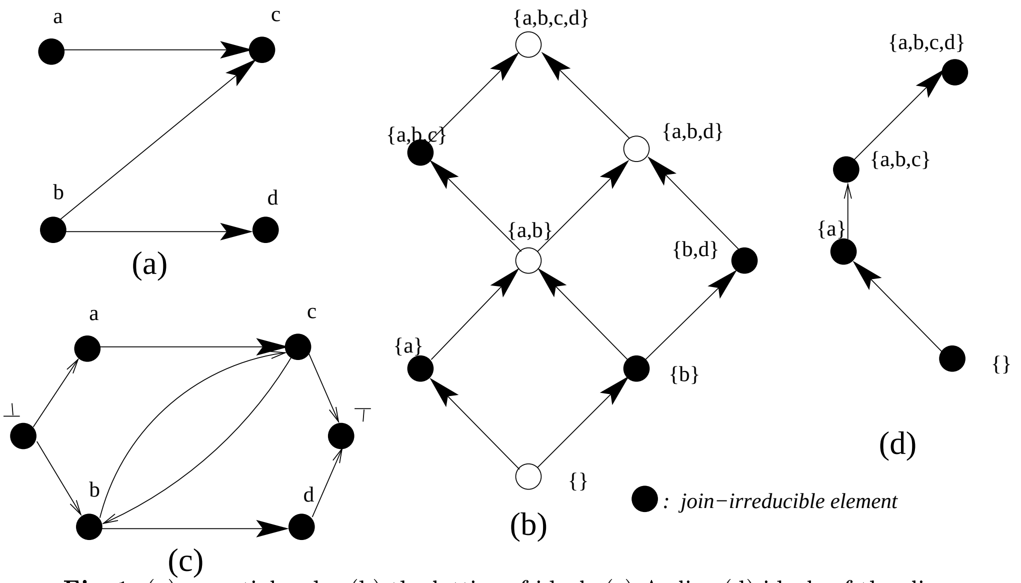

Fig. 1. (a) a partial order (b) the lattice of ideals (c) A slice (d) ideals of the slice.

manner. $M(f)$, the greatest ideal that does not contain f, is meet-irreducible. The set of all meet-irreducible elements of L are denoted by $M(L)$ and Birkhoff's theorem can also be stated using $M(L)$.

In this paper, P and posets derived from P will serve as the small structures, and L and sublattices of L will serve as the large structures. We are usually interested in $L_B \subseteq L$, containing ideals of L that satisfy a given predicate B. Instead of enumerating L and checking for predicate B, we use P and B to derive a structure P_B that generates L_B.

For counting the number of elements in L and its sublattices, we use $N(P)$ to denote the number of ideals of the poset P. Since our interest is in efficient calculation of $N(P)$, we will restrict the *dimension* of the partial order generating the lattice. The dimension of a poset (X, P), denoted by $dim(X, P)$, is the least positive integer t for which there exists a family $\{C_1, C_2, \ldots, C_t\}$ of linear extensions of P (total orders compatible with P) such that $P = \cap_{i=1}^{t} C_i$. Determining whether a poset P with n points is 2-dimensional and isomorphism testing for 2-dimensional orders can be done in $O(n^2)$ time [Spi85]. All the posets used in this paper are 2-dimensional. The reader is referred to [Tro92] for dimension theory of posets. The following lemma shows that the number of ideals of a poset can be calculated efficiently for series-parallel posets (a special case of 2-dimensional posets) [FLST86]. For generalization to counting ideals of two dimensional posets see [Ste84].

Lemma 1 (Counting Lemma). *(1) If Q is an extension of P then $N(Q) \leq N(P)$. (2) Let $P + Q$ be the disjoint union (or direct sum) of posets P and Q (see [DP90]). Then, $N(P + Q) = N(P)N(Q)$. (3) Let $P \oplus Q$ be the ordinal sum of posets P and Q[DP90]. Then, $N(P \oplus Q) = N(P) + N(Q) - 1$. (4) Assume that P can be decomposed into the least number of chains $C_1, C_2, \ldots C_n$. Then $N(P) \leq \prod_{i=1}^{n}(|C_i| + 1)$. When each chain is at most m in length, we get that $N(P) \leq (m + 1)^n$.*

For some examples, instead of enumerating all ideals of a poset we may be interested in enumerating or counting ideals in a certain *level set*. To define *level sets*, first define a poset to be *ranked* if for each element $x \in P$, one can assign a non-negative integer, $rank(x)$, such that if y covers x, then $rank(y) = rank(x) + 1$. The set of all elements in P with rank i are called it level set with rank i. Every distributive lattice is a ranked poset [Sta86].

3 Our Model

Traditionally the duality is expressed between finite posets and finite distributive lattices. In this paper, we are interested in producing structures that generate subsets of the finite distributive lattice. It is more convenient to use directed graphs instead of posets for this purpose because, as shown later, we can get sublattices by simply adding edges to the original directed graph.

The notion of ideals can be extended to graphs in a straightforward manner. A subset of vertices of a directed graph is an *ideal* iff the subset contains a vertex only if it contains all its incoming neighbors. Observe that an ideal either contains all vertices in a strongly connected component or none of them. Let $\mathcal{I}(P)$ denote the set of ideals of a directed graph P. Observe that the empty set $\emptyset$ and the set of all vertices trivially belong to $\mathcal{I}(P)$. We call them *trivial* ideals. It is shown in [MG01] that given a directed graph P, $\langle \mathcal{I}(P); \subseteq \rangle$ forms a distributive lattice.

Now assume that we are interested in the set of ideals that satisfy a predicate B. We will be interested in deriving a graph such that its ideals capture this set. A small difficulty is that every graph has at least two trivial ideals and therefore we cannot capture the case when the set of ideals satisfying B is empty. To address this problem, we add to the graph two additional vertices $\bot$ and $\top$ such that $\bot$ is the smallest vertex and $\top$ is the largest vertex. This ensures that any nontrivial ideal will contain $\bot$ and will not contain $\top$. As a result, every ideal of a computation in the traditional model is a nontrivial ideal of the computation in our model and vice versa. We will deal with only nontrivial ideals from now on.

Fig. 1(c) shows the directed graph and its nontrivial ideals. The directed graph in Fig. 1(c) is derived from Fig. 1(a) by adding an edge from c to b and adding two additional vertices $\bot$ and $\top$. The resulting set of nontrivial ideals is a sublattice of Fig. 1(d). In the figure, we have not shown $\bot$ in the ideals because it is implicitly included in every nontrivial ideal. This example illustrates the main steps in our technique. Fig. 1(a) is a small structure that generates the large structure Fig. 1(b). We are interested in enumerating or counting the set of ideals that satisfy the property "the ideal contains b whenever it contains c." To generate such ideals it is sufficient to add an edge from c to b. Fig. 1(c) shows the small structure that generates all the ideals of interest to us. Fig. 1(c) will be called the slice of Fig. 1(a) with respect to the predicate B. The formal definition of a slice is given in the next section.

4 Slices and Regular Predicates

We denote the slice of a directed graph P with respect to a predicate B by $slice(P, B)$. The $slice(P, B)$ is a graph derived from P such that all the ideals in $\mathcal{I}(P)$ that satisfy B are included in $\mathcal{I}(slice(P, B))$. Note that the slice may include some additional ideals which do not satisfy the predicate. Formally,

Definition 1 (Slice [MG01]). *A slice of a graph P with respect to a predicate B is the directed graph obtained from P by adding edges such that (1) it contains all the ideals of P that satisfy B and (2) of all the graphs that satisfy (1), it has the least number of ideals.*
It is shown in [MG01] that the slice exists and is unique for every predicate. Computing slices for predicates in general is NP-hard but one can efficiently compute slices for *regular* predicates.

Definition 2 (Regular Predicates [GM01]). *A predicate is regular if the set of ideals that satisfy the predicate forms a sublattice of the lattice of ideals.*

Equivalently, a predicate B is *regular* with respect to P if it is closed under $\sqcup$ and $\sqcap$.

We now show that regular predicates can be decomposed into simpler structures called *simple* predicates. Our motivation is that computing slices for simple predicates is easy.

Definition 3 (Simple Predicates). *A predicate B is* simple *if there exists $e, f \in P$ such that $\forall G \in \mathcal{I}(P): \quad B(G) \equiv ((f \in G) \Rightarrow (e \in G))$*
Denote this predicate by $S(e, f)$. Thus, a simple predicate is of the form: G satisfies B iff whenever it includes f it includes e. We first show a useful property of simple predicates.

Lemma 2. *A simple predicate $S(e, f)$ partitions the lattice of ideals into two sublattices. Moreover, $\neg S(e, f)$ is equivalent to the interval lattice $[J(f), M(e)]$.*

In Fig. 1(c), our predicate is $S(c, b)$. The sublattice for $S(c, b)$ is shown in Fig. 1(d). Its complement, the set of ideals $[\ \{b\}, \{a, b\}, \{b, d\}, \{a, b, d\}\ ]$ is also a sublattice. We now show an easy test that indicates whether a regular predicate B is stronger than $S(e, f)$. Let $J_B(e)$ denote the least ideal that includes e and satisfies B. Since the predicate B is regular and the predicate "the ideal includes e" is also regular, it follows that $J_B(e)$ is well defined.

Lemma 3. *For regular B and any e,f*
$e \in J_B(f) \quad \equiv \quad J_B(e) \subseteq J_B(f) \quad \equiv \quad B \Rightarrow S(e, f)$

We now turn our attention to characterizing the set of ideals that satisfy B.

Lemma 4. *An ideal G satisfies a regular predicate B iff $\forall f \in G : J_B(f) \subseteq G$.*

We now provide a *decomposition* theorem for regular predicates.

Theorem 1. *For any regular predicate B, let $E_B = \{(e, f) | B \Rightarrow S(e, f)\}$. Then, $B = \bigwedge_{(e,f)\in E_B} S(e, f)$.*

From the decomposition theorem and properties of simple predicates we get that B is a regular predicate *iff* it can be expressed as a conjunction of simple predicates. As a corollary (by applying De Morgan's and using the result about complement of simple predicates), we also get the following Rival's theorem [Riv73].

Corollary 1. *A complement of a sublattice can be expressed as a union of interval lattices of the form $[c, d]$ where c is a join-irreducible element and d is a meet-irreducible element.*

This also implies that there are $O(2^{n^2})$ distributive lattices on n points. Every distributive lattice is a sublattice of the boolean lattice on n elements and therefore equivalent to a regular predicate. By the decomposition theorem, a regular predicate is a conjunction of at most $O(n^2)$ simple predicates. Note that there can be as many as $O(2^{2^n})$ subsets of the boolean lattice but only very few of them are sublattices.

Now obtaining slices for a regular predicate B is an easy task. We simply add edge (e, f) to the graph of P for every simple predicate $S(e, f)$ such that $B \Rightarrow S(e, f)$. Therefore, we have

Theorem 2. *Let P be a directed graph. Let Q be a directed graph obtained by adding edges to P. Then, $\mathcal{I}(Q)$ is a sublattice of $\mathcal{I}(P)$. Conversely, every sublattice of $\mathcal{I}(P)$ is generated by some directed graph Q obtained from P by adding edges.*

Suppose that the poset P has n chains each of size at most m. Observation that if $f \leq g$ in poset P, then for any e, $S(e, f)$ implies $S(e, g)$. Therefore, for any event e there are at most n simple predicates (at most one for every chain) as part of a regular predicate. We conclude that every regular predicate can be expressed as conjunction of at most n^2m simple predicates.

5 Application to Combinatorics

In this section we give several examples of slicing posets.

1. Boolean Algebra and Set Families

Let X be a ground set on n elements. By using $\subseteq$ as the order relation on the power set of X, we can view it as a distributive lattice L. This lattice (called boolean lattice) has $n + 1$ levels and each level set of rank k in the boolean lattice corresponds to $\binom{n}{k}$ sets of size k. L is generated by the directed graph in Fig. 2(a) which can also be interpreted as a distributed computation n processes $\{P_1, \ldots P_n\}$. Each process P_i executes a single event e_i. It is easy to verify that there is a bijection between every nontrivial global state of the computation and a subset of X.

Now consider all subsets of X such that if they include e_i then they also include e_j. To obtain the slice with respect to this predicate we just need to add an edge from e_j to e_i. Fig. 2(b) shows the slice with respect to the predicate that if e_3 is included then so is e_2. To ensure the condition that e_i is always included,

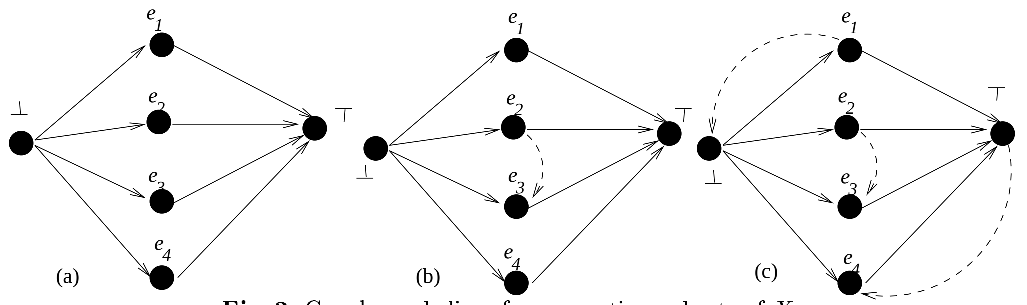

Fig. 2. Graphs and slices for generating subsets of X

we simply add an edge from e_i to $\perp$ and to ensure that e_i is never included in any subset, we add an edge from $\top$ to e_i. Fig. 2(c) shows the slice which gives all subsets that always contain e_1, never contain e_4 and contain e_2 whenever they contain e_3.

As an application, we now solve some combinatorial problems. Let n be even. We are required to calculate the total number of subsets of $[n]$ which satisfy the property that if they contain any odd integer i, then they also contain $i+1$ (or equivalently, compute the number of ways to select groups from $n/2$ couples such that a wife is always accompanied by her husband in the group although a husband may not be accompanied by his wife). Although this problem can be solved directly by a combinatorial argument, we show how our method can be applied. First construct the poset which generates all the subsets of $[n]$. It is Fig. 2(a) in this case. Now define the subset of interest by a predicate B. For any subset G of $[n]$, Let $B(G)$ to be true if G contain $i+1$ whenever it contains any odd integer i. From our discussion of regular predicates, it is clear that B is regular and equivalent to $S(e_2, e_1) \wedge S(e_4, e_3) \ldots \wedge S(e_n, e_{n-1})$. To compute the slice, it is sufficient to add an edge from e_{i+1} to e_i for odd i. The slice consists of $n/2$ chains each with exactly 2 events (ignoring $\perp$ and $\top$). From the counting lemma (Lemma 1), it follows that the total number of ideals is $(2+1)^{n/2} = 3^{n/2}$. The reader should note that for any fixed value of n, the problem can be solved by a computer automatically and efficiently (because the slice results in a series-parallel poset).

2. Set families of Size k

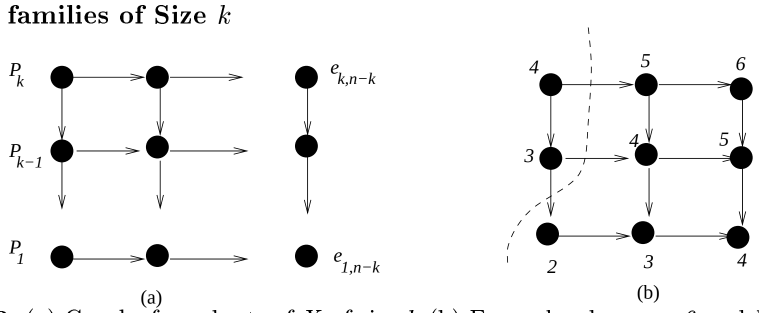

Fig. 3. (a) Graphs for subsets of X of size k (b) Example when $n = 6$ and $k = 3$

It is important to note that regularity of B is dependent upon the lattice structure of L. For example, in many applications of set families, we are interested in sets of a fixed size k. The predicate B that the ideal is of size k is not regular. However, by considering alternative posets, this set family can still be analyzed. Fig. 3 shows a computation such that all the subsets of X of size k are its ideals. For clarity, we have not drawn $\top$ and $\bot$ in the figure.

There are k processes in this computation and each process executes $n-k$ events. By the structure of the computation, if in a global state P_i has executed j events, then P_{i+1} must have also executed at least j events. The correspondence between subsets of X and global states can be understood as follows. If process P_i has executed t events in the global state, then the element $t+i$ is in the set Y. Thus process P_1 chooses a number from $1 \ldots n-k+1$ (because there are $n-k$ events); process P_2 chooses the next larger number and so on. It can also be easily verified that the poset in Fig. 3(a) is a 2-dimensional poset and that there are $\binom{n}{k}$ ideals of this poset. Fig. 3 gives an example of the computation for subsets of size 3 of the set [6]. The global state, or the ideal, shown corresponds to the subset $\{1,3,4\}$.

Now let us apply our theory to the first combinatorial problem (Q1) mentioned in the introduction. Assume that we are interested in counting all subsets of n of size k which do not have any consecutive numbers. In this example, G satisfies B if whenever P_i has t events in G, P_{i+1} has at least $t+1$ events in G. This condition is regular and we can use Lemma 3 and Theorem 1 to compute the slice. (for every event f, we only need to determine whether $e \in J_B(f)$). Fig. 4 shows the slice which includes precisely such subsets. By collapsing all strongly connected components and by removing the transitively implied edges we get a graph which is isomorphic to the case when there are k processes and each process executes $n-k-(k-1)$ events. Therefore, the total number of such sets is $\binom{n-k+1}{k}$. Again one can come up with a combinatorial argument to solve the problem (for example, see Theorem 13.1 and Example 13.1 in [vLW92]), but the slicing approach is completely mechanical.

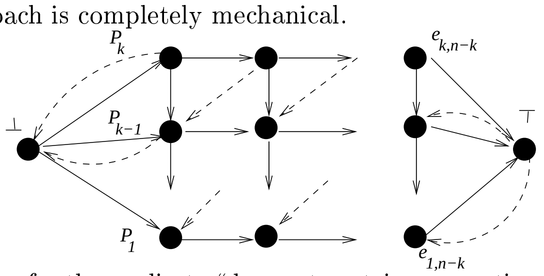

Fig. 4. Slice for the predicate "does not contain consecutive numbers"

The above construction can be generalized to multidimensional grids to obtain results on multinomials instead of binomials.

3. Integer Partitions and Young's lattice

A k-tuple of positive integers $\lambda = (\lambda_1, \ldots, \lambda_k)$ is an integer partition of N if $\lambda_1 + \ldots + \lambda_k = N$ and for all i, $\lambda_i \geq \lambda_{i+1}$. The number of *parts* of λ is k. An example of partition of 10 into 3 parts is $(4,3,3)$. An integer partition can be visualized as a *Ferrers diagram* or an array of squares in decreasing order with λ_i squares in row i. The Ferrers diagram of the partition $(4,3,3)$ of 10 is shown in Fig. 5(a). A partition λ is contained in another partition δ if the number of parts of λ is at most that of δ and λ_i is less than or equal to δ_i for any i between 1 and the number of parts of λ. For example, $(3,3,1)$ is less than $(4,3,3)$. Fix any partition λ. The set of all partitions that are less than or equal to λ form the *Young's lattice* denoted by Y_λ.

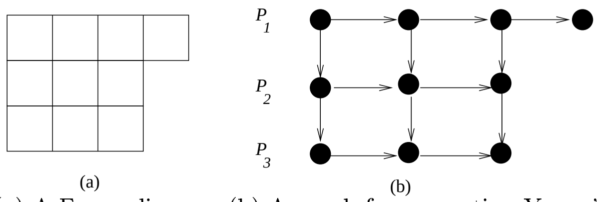

Fig. 5. (a) A Ferrer diagram (b) A graph for generating Young's lattice

We now apply our approach to Y_λ. Let the number of parts and the largest part in the partition λ be m and n respectively. Then we have a distributed computation of n processes with at most m events per process as shown in Fig. 5(b). P_i executes as many events as λ_i. It is clear that for any global state, the number of events executed by P_i is at least as many as executed by P_{i+1}. Clearly, the set of global states of the computation as in Fig. 5(b) is isomorphic to Young's lattice for the corresponding partition.

It follows that Young's lattice is distributive. One can see that the lattice of subsets of size k from the set of size n is a special case of Youngs's lattice when all λ_i's are equal. Therefore, the number of integer partitions whose Ferrers diagrams fit in a box of size k by $n-k$ is equal to $\binom{n}{k}$ (providing an alternate proof of Theorem 3.2 in [SW86]). Let $q(N,k,m)$ denote the number of partitions of N whose Ferrer's diagram fit in a box of size k by m. By summing up the sizes of all level sets, we get $\binom{n}{k} = \sum_{l=0}^{k(n-k)} q(l,k,n-k)$. Since the poset that generates corresponding Young's lattice is symmetric with respect to k and m, we get that $q(N,k,m)$ equals $q(N,m,k)$; and since the poset is dual of itself (i.e. we get back the same poset when all arcs are reversed) we also get that $q(N,k,m)$ equals $q(mk-N,k,m)$. All these results are well known and generally derived using Gaussian polynomials (see [vLW92]).

Now assume that we are interested in all those partitions such that their second component is some fixed value say b. It is easy to verify that partitions $\delta \in Y_\lambda$ such that $\delta_c = b$ form a sublattice and therefore the condition $\delta_c = b$ is a regular predicate. Fig. 6(a) gives the slice of partitions in which $\delta_2 = 2$. Since the second part must be 2, we add edges to ensure that P_2 executes exactly 2 events. On collapsing the strongly connected components, transitively reducing the graph and applying counting lemma, we get that there are $(2+1)(2+1) = 9$ such partitions.

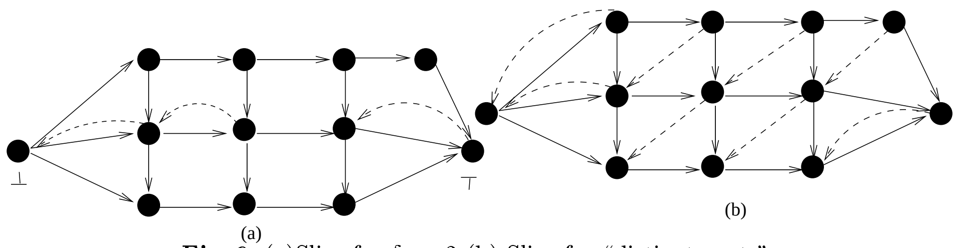

Fig. 6. (a)Slice for $\delta_2 = 2$ (b) Slice for "distinct parts"

As another example assume that we are interested in all partitions less than λ which have distinct parts. Fig. 6(b) gives the slice. The graph is equivalent to that of subsets of size 3 from [5]. Hence, there are $\binom{5}{3}$ such partitions. Some other subsets of partitions discussed in the literature are "partitions with odd number of parts", "partitions with distinct odd parts," "partitions with even number of parts" etc. These are also regular predicates.

Now the reader may also see the solution for the second problem (Q2) mentioned in the introduction—enumerating all partitions in the Young's lattice Y_λ with first part equal to the second part. We simply define the predicate B on a partition δ to be true when δ_1 equals δ_2. It is clear that the predicate is closed under joins and meets and is therefore a regular predicate. One can draw the slice and conclude that the number of partitions δ in Y_λ satisfying $\delta_1 = \delta_2$ is equal to the number of partitions in Y_δ where $\delta = (\lambda_2, \lambda_3, \ldots, \lambda_k)$.

Note that the level set of rank N of Y_λ (where $\lambda = (\lambda_1, \lambda_2 \ldots, \lambda_t)$) corresponds to all partitions of N with at most t parts and the largest part at most λ_1. It follows that all partitions of N can be enumerated as the elements in level set of rank N of $Y_{(N,N,..N)}$.

4. Permutations

We first show a small computation that generates all permutations of n symbols. The computation consists of $n-1$ processes. Process P_i executes $i-1$ events. We use the notion of the inversion table[Knu98] to interpret the choices made by processes. The number of inversions of i in a permutation π is the number of symbols less than i that appear to the right of i in π. The way a permutation is generated from a global state is as follows. We begin the permutation by writing 1. P_1 decides where to insert the symbol 2. There are two choices. If we place 2 after 1, then we introduce zero inversions; otherwise we introduce one inversion. Proceeding in this manner we get that there is a bijection between the set of permutations and the global states.

It is easy to show that

Lemma 5. *All the following properties of permutations are regular. (1) The symbol $m < n$ has* at most *j inversions (for $j < m$). The total number of such permutations is $\frac{n!(j+1)}{m}$. (2) $i \leq j$ implies that i has at most as many inversions as j. The total number of such permutations is same as the number of integer partitions less than $(n-1, n-2, ..., 1)$.*

Further by computing the slice, we can also calculate the number of permutations satisfying B. The level set at rank k of the permutation lattice consists of all permutations with total number of inversions equal to k and therefore such permutations can be efficiently enumerated [Knu98,ER02].

References

[AV01] S. Alagar and S. Venkatesan. Techniques to tackle state explosion in global predicate detection. *IEEE Transactions on Software Engineering*, 27(8):704 – 714, August 2001.

[CM91] R. Cooper and K. Marzullo. Consistent detection of global predicates. In *Proc. of the Workshop on Parallel and Distributed Debugging*, pages 163–173, Santa Cruz, CA, May 1991. ACM/ONR.

[DP90] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order.* Cambridge University Press, Cambridge, UK, 1990.

[ER02] S. Effler and F. Ruskey. A CAT algorithm for listing permutations with a given number of inversions. *Information Processing Letters*, 2002.

[FLST86] U. Faigle, L. Lovász, R. Schrader, and Gy. Turán. Searching in trees, series-parallel and interval orders. *SIAM Journal on Computing*, 15(4):1075–1084, 1986.

[Gar02] V. K. Garg. Detecting global predicates in distributed computations. Technical report, Parallel and Distributed Systems Laboratory, ECE Dept. University of Texas at Austin, September 2002. available at `www.ece.utexas.edu/~garg/pubs.html`.

[GM01] V. K. Garg and N. Mittal. On slicing a distributed computation. In *21st International Conference on Distributed Computing Systems (ICDCS' 01)*, pages 322–329, Washington - Brussels - Tokyo, April 2001. IEEE.

[Knu98] Donald E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming.* Addison-Wesley, Reading, MA, USA, second edition, 1998.

[MG01] N. Mittal and V. K. Garg. Slicing a distributed computation: Techniques and theory. In *5th International Symposium on DIStributed Computing (DISC'01)*, pages 78 – 92, October 2001.

[PB83] J.S. Provan and M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12:777–788, 1983.

[Riv73] I. Rival. Maximal sublattices of finite distributive lattices. *Proc. Amer. Math. Soc.*, pages 417–420, 1973.

[Spi85] Jeremy Spinrad. On comparability and permutation graphs. *SIAM Journal on Computing*, 14(3):658–670, 1985.

[Squ95] M. Squire. *Gray Codes and Efficient Generation of Combinatorial Structures.* PhD Dissertation, Department of Computer Science, North Carolina State University, 1995.

[Sta86] R. Stanley. *Enumerative Combinatorics Volume 1.* Wadsworth and Brookes/Cole, Monterey, California, 1986.

[Ste84] G. Steiner. Single machine scheduling with precedence constraints of dimension 2. *Math. Operations Research*, 9:248 – 259, 1984.

[Ste86] G. Steiner. An algorithm to generate the ideals of a partial order. *Operations Research Letters*, 5(6):317 – 320, 1986.

[SW86] D. Stanton and D. White. *Constructive Combinatorics.* Springer-Verlag, 1986.

[Tro92] W.T. Trotter. *Combinatorics and Partially Ordered Sets: Dimension Theory.* The Johns Hopkins University Press, 1992.

[vLW92] J.H. van Lint and R. M. Wilson. *A Course in Combinatorics.* Cambridge University Press, 1992.