COMBINATORICA 7 (4) (1987) 365—374

RANDOMIZED ROUNDING: A TECHNIQUE FOR PROVABLY GOOD ALGORITHMS AND ALGORITHMIC PROOFS

P. RAGHAVAN and C. D. THOMPSON

Received 1 September 1985
Revised 1 May 1986

We study the relation between a class of 0—1 integer linear programs and their rational relaxations. We give a randomized algorithm for transforming an optimal solution of a relaxed problem into a provably good solution for the 0—1 problem. Our technique can be a of extended to provide bounds on the disparity between the rational and 0—1 optima for a given problem instance.

1. General Outline

The relation of an integer program to its rational relaxation has been the subject of considerable interest [1], [5], [11]. Such efforts fall into two categories: (1) Showing existence results for feasible solutions to an integer program in terms of the solution to its rational relaxation, and (2) Using the information derived from the solution of the relaxed problem in order to construct a provably good solution to the original integer program.

We present a technique here which we call *randomized rounding*. This technique is applicable to a class of 0—1 integer linear programs, and yields results in both the categories listed above. Our technique is probabilistic; for the existence results, we prove that the solution to an integer program satisfies a certain property by showing that a randomly generated solution satisfies that property with non-zero probability. In this random generation of solutions, we make use of the optimal solution to the rational relaxation linear program. By modifying the procedure used to derive the existence result, we can obtain an algorithm that is *provably good* in the following sense. We show that with high probability, our algorithm will provide an integer solution in which the objective function takes on a value close to the optimum of the rational relaxation. This is a sufficient condition to show the near-optimality of our 0—1 solution since the optimal value of the objective function in the relaxed version is no worse than the optimal value of the objective function in the original 0—1 integer program.

This work was supported by Semiconductor Research Corporation grant SRC 82-11-008 and an IBM Doctoral Fellowship.

AMS subject classification: 90 C 10.

We now give a general outline of the technique. Let Π_I be a 0—1 linear program, with variables $x_i \in \{0, 1\}$. Let Π_R be its rational relaxation, with $x_i \in [0, 1]$. The basic algorithm consists of the following two phases:
(1) Solve Π_R; let the variables take on values $\hat{x}_i \in [0, 1]$.
(2) Set the variables x_i randomly to one or zero according to the following rule:

$$\text{Prob. } [x_i = 1] = \hat{x}_i, \tag{1.1}$$

In proving results about the outcome of our algorithm, we repeatedly make use of the following results from probability theory. Let $B(m, N, p)$ denote the probability that there will be at least m successes in N Bernoulli trials each with success probability p.

Theorem 1.1. (Hoeffding) [6]. *If $\Psi_1, \Psi_2, \ldots, \Psi_N$ are completely independent Bernoulli trials such that $E(\Psi_k)=p_k$ and $\Psi=\Psi_1+\Psi_2+\ldots+\Psi_N$, we have*

$$\text{Prob. } (\Psi \geqq m) \leqq B(m, N, p) \tag{1.2}$$

where

$$p = \frac{\sum_{k=1}^{N} p_k}{N}. \quad \blacksquare$$

The other fact that we require is a bound on the tail of the binomial distribution due to Angluin and Valiant [2], based on a general technique due to H. Chernoff [3]:

Theorem 1.2. *If $m=(1+\beta)\cdot Np$, then for $0<\beta\leqq 1$,*

$$B(m, N, p) < \exp\left(-\frac{\beta^2 Np}{3}\right). \tag{1.3}$$

Remark. *The inequality can be shown to be strict provided $Np>0$.*

In the next two sections, we provide some direct applications of the technique: section 2 deals with a routing problem that arises in the design of VLSI circuits, and the following section treats the 0—1 multicommodity flow problem. Section 4 provides an extension to the basic technique in order to deal with some situations that cannot be directly handled. The problem of *simple k-matching* is used to illustrate this extension, which we call *scaling*. Section 5 concludes with remarks on whether our bounds can be improved.

2. A Routing Problem im VLSI

In this section we illustrate the basic principles of the randomized rounding technique by means of a routing problem that arises in the design of a certain class of VLSI circuits. The problem is that of global routing in gate-arrays [14], and is defined as follows.

We are given a two-dimensional rectilinear $n\times n$ lattice L_n. In the context of gate-arrays, lattice-nodes represent logic circuit elements and lattice-edges represent *channels* in which wires used to connect the nodes can be routed. In an instance of a global routing problem, we are given a collection of *nets*, where a net a_i is a set of nodes to be connected by means of a Steiner tree in L_n. In addition, for each net a_i, we are given a set of possible trees b_{ij} that can be used for connecting the nodes in that net. A solution to the problem consists of choosing one tree for each net in the instance, from the allowed possibilities for that net. The number of trees in a solution that contain a given lattice edge is termed the *width* of that edge in that solution. The *width* of a solution is the maximum width of an edge taken over all edges in the lattice. Our objective is to find a solution of minimum width.

This problem is readily formulated as a 0—1 integer program by assigning a variable for each configuration of each net: thus, let x_{ij} be an indicator variable denoting whether or not the j^{th} tree b_{ij} is chosen for net a_i. Constraints of the form

$$\sum_j x_{ij} = 1, \quad \forall i \tag{2.1}$$

ensure that a choice is made for each net. The number of trees in the solution that contain a given edge e is bounded above by an unknown quantity W which we seek to minimize in an objective function. We express these constraints by means of

$$\sum_{b_{ij} \text{ contains } e} x_{ij} \leqq W, \quad \forall e \tag{2.2}$$

and

$$\text{Minimize } W, \text{ s.t. (2.1), (2.2) and } x_{ij} \in \{0, 1\} \quad \forall i, j. \tag{2.3}$$

The formulation above is similar to formulations due to Hu and Shing [7] and Karp *et al.* [10]. Consider a linear programming relaxation of (2.3) in which fractional solutions are allowed:

$$\text{Minimize } W, \text{ s.t. (2.1), (2.2) and } x_{ij} \in [0, 1] \quad \forall i, j. \tag{2.3a}$$

The optimum solution to (2.3a) can be found in polynomial time [8]. Let the optimum (fractional) value of x_{ij} be $\hat{x}_{ij}$. Furthermore, let W_1 be the optimum width obtained from the linear program solution; W_1 is a lower bound on the best possible integer optimum width. We now seek to use these fractional solutions to obtain integer solutions (2.3). We do this by means of randomization: for each i, set x_{ij} to one with probability $\hat{x}_{ij}$. The choice is done in an exclusive manner, with constraint (2.1): for each i, exactly one of the x_{ij} is set to one; the rest are set to zero. This random choice is made independently for all i.

Theorem 2.1. *Let ε be a positive real such that $0<\varepsilon<1$. Provided*

$$W_1 \geqq 3 \ln (2n(n-1)/\varepsilon),$$

the width of the solution produced by the above procedure does not exceed

$$W_1 + \left(3 \cdot W_1 \cdot \ln \frac{2n(n-1)}{\varepsilon}\right)^{1/2} \tag{2.4}$$

with probability at least $1-\varepsilon$.

Proof. The proof follows from the observation that the width of a latticeedge e is the sum of independent Bernoulli trials. The expected value of this sum is no more than W_1, since the biases used for the coin-flipping were the $\hat{x}_{ij}$ determined by the LP. Hoeffding's lemma is thus applicable with $p=W_1/N$. Chernoff's bound is now applied with

$$\beta \geqq \left[\frac{3\ln\dfrac{2n(n-1)}{\varepsilon}}{W_1}\right]^{1/2}. \tag{2.5}$$

This ensures that the rounded width of any edge does not exceed the upper bound in (2.4) with probability at least $1-\varepsilon/(2n(n-1))$; then the maximum of the widths of the $2n(n-1)$ edges in the lattice does not exceed (2.4) with probability $1-\varepsilon$. ∎

The second (randomization) stage can be repeated to improve the solution. In $n\times n$ gate-arrays, W_1 grows as n^c [13] for some $c\in(0.5,1]$. The approximation of Theorem 2.1 is thus asymptotically a good one in that our solution is weaker than the best possible by an additive term.

Theorem 2.1 gives (probabilistically) a provably good solution to the routing problem. Viewed in a slightly different light, it is also a proof that there exists an integer solution to (2.3) whose objective function value is related to W_1 by the following relation.

Theorem 2.2. *Let* $W_1>3\ln 2n(n-1)$ *be the optimum objective function value of the linear programming relaxation of* (2.3). *Then, there exists an integer solution of width not exceeding*

$$W_1+(3\cdot W_1\cdot\ln 2n(n-1))^{1/2}. \tag{2.6}$$

Proof. Similar to the proof of Theorem 2.1. ∎

3. Undirected Multicommodity Flow Problems

In undirected multicommodity flow problems, we are given an undirected graph $G(V, E)$. In an instance of the problem, various vertices are the sites of *sources* and *sinks* (sources are denoted by s_i and sinks by t_i, $1\leqq i\leqq k$). A vertex $v\in V$ may be the location of more than one source (sink). One unit of flow is to be conveyed from each source s_i to its corresponding sink t_i through the edges in E. Each edge $e\in E$ has a capacity $c(e)$ which is an upper limit on the total amount of flow in e. We insist that the flow of any commodity in any edge be either zero or one. Note that an edge could have flow going in both directions; for instance, the flow from s_i to t_i (hereafter referred to as the flow of commodity i) could be in a direction opposite to that of the flow of commodity j in some edge e. Each of these commodities uses up one unit of the capacity of that edge, regardless of their direction.

We consider two types of such multicommodity flow problems. In the first kind, we try to maximize the total flow subject to meeting the capacity constraints (as well as conservation constraints for each commodity at each vertex). In a second variant of the problem, we require that all edges must have the same capacity; we try to minimize this common capacity while realizing unit flows for all k commodities. In this section, we focus on the second variant; the techniques used in its solution, together with some new ones to be introduced in the next section, can be used to solve the first variant. The general integral problem is known to be NP-Complete [4], although the non-integral version can be solved using linear programming methods [9] in polynomial time.

The algorithm consists of the following three major phases:

(1) Solving a non-integral multi-commodity flow problem.
(2) Path stripping.
(3) Randomized path selection.

Non-integral Muticommodity Flow: As in the previous section, we relax the requirement of 0—1 flows to allow fractional flows in the interval [0,1]. The relaxed capacity-minimization problem can be solved, for instance, by linear programming. Let us then assume that we have solved the non-integral problem and assigned to each edge $e \in E$ a flow $f_i(e) \in [0, 1]$ for each commodity i. A capacity constraint of the form

$$\sum_{i=1}^{k} f_i(e) \leq C \tag{3.1}$$

is then satisfied for each $e \in E$, where C is the optimal solution to our nonintegral edge-capacity optimization problem. As before, C is a lower bound on the best possible integral solution.

Path stripping: The main idea of this phase is to convert the edge flows for each commodity i into a set Γ_i of possible paths which could be used to realize the flow of that commodity. Initially, Γ_i is empty.

For each i:

(1) Form a directed graph $G_i(V, E_i)$ where E_i is a set of directed edges derived from E as follows: For each $e \in E$, assign a direction to e which is the direction of flow of commodity i in e. If $f_i(e)=0$, e is excluded from E_i.
(2) Discover a directed path $\{e_1, \ldots, e_p\}$ in G_i from s_i to t_i using a depthfirst search, discarding loops. Let

$$f_m = \min \{f_i(e_j), \quad 1 \leq j \leq p\}. \tag{3.2}$$

For $1 \leq j \leq p$, replace $f_i(e_j)$ by $f_i(e_j) - f_m$. Add the path $\{e_1, \ldots, e_p\}$ to Γ_i along with its *weight* f_m.
(3) Remove any edges with zero flow from E_i. If there is non-zero flow leaving s_i, repeat step (2). Otherwise, next i.

It is clear that the above process terminates, since at each execution of step (2), at least one edge (the one with minimum flow in the path) is deleted from E_i. Thus the number of times it is executed is upper bounded by $|E_i|$. It is also evident that on termination, the sum of the weights of the paths in Γ_i is one.

The idea of path-stripping is similar to one used in the network-flow algorithm of Malhotra *et al.* [12]. Their method of finding a blocking flow in a layered network consists of successively saturating vertices of minimum throughput.

Randomization: **For each** i:

Cast a $|\Gamma_i|$-faced die with face-probabilities equal to the weights of the paths in Γ_i. Assign to commodity i the path whose face comes up. Next i.

We can then prove a theorem similar to Theorem 2.1:

Theorem 3.1. Let ε be a positive real such that $0<\varepsilon<1$. Provided $C \geqq 2 \ln |E|$, the integer capacity of the solution produced by the above procedure does not exceed

$$C+\left(3 \cdot C \cdot \ln \frac{|E|}{\varepsilon}\right)^{1/2} \tag{3.3}$$

with probability at least $1-\varepsilon$.

Proof. The proof is similar to that of Theorem 2.1, invoking Hoeffding's and Chernoff's inequalities. The expected number of unit flows through edge e is given by (3.1). ∎

An existence result similar to Theorem 2.2 can be inferred readily from the above theorem. In [14] it is shown that the path-stripping and randomization phases described above can be replaced by a random-walk, with the same results.

4. Randomized Rounding with Scaling

The problems considered in the previous sections were similar in that the right-hand sides of the major constraints were the objective function itself (W in the routing problem and C in the flow problem). In this section we will consider the case when the right-hand sides of the constraints defining the problem are parameters independent of the objective function. A new technique which we call *scaling* is introduced in order to handle such problems.

Let k be a fixed quantity. Consider a constraint of the form

$$\sum_i x_i \leqq k. \tag{4.1}$$

Moreover, suppose that we have fractional values $\hat{x}_i$ for these variables (derived from the solution of the appropriate relaxation), and the $\hat{x}_i$ are then interpreted as probabilities for a randomized rounding phase as in the previous sections. The difficulty lies in the fact that there is a significant probability that the values of the x_i after rounding will not satisfy (4.1). Furthermore, it is not clear whether there is a non-zero probability that the randomized rounding will yield a solution in which none of n constraints is violated. We now present a device by which we can reduce the probability that a constraint is violated to less than $1/n$. We call this device scaling. In this manner, we reduce the probability that *some* constraint is violated to less than one.

The idea is to multiply each of the $\hat{x}_i$ by some fraction less than one. The resultant value is used in the rounding stage as the probability that x_i is set to one. Intuitively, this reduces the number of variables that are set to one and thus the probability that a constraint is violated. The example below illustrates the scaling technique, together with the details of determining the fraction used. We consider the problem of *simple k-matching* defined below. We use the terminology of Lovász [11].

A *hypergraph H* is a finite set of *edges*, where an edge is a non-empty subset of an n-element set V. The elements of V are called *vertices*. A *k-matching* of H is a set

M of edges such that each vertex in V belongs to at most k of the edges in M. The maximum number of edges in any k-matching of H is denoted by $v_k(H)$. A k-matching is *simple* if no edge of H occurs more than once in M. The maximum number of edges in any simple k-matching of H is denoted by $\tilde{v}_k(H)$. The problem of determining $\tilde{v}_k(H)$ can be formulated as an integer program as follows.

Suppose H has n vertices and m edges. Let $\mathbf{A}$ be an $n\times m$ matrix in which all the entries are either zero or one; $\mathbf{A}$ represents the vertex-edge incidence matrix of H. Let x_i, $1\leq i\leq m$ be 0—1 indicator variables that denote whether or not edge i is in M. Let $\mathbf{x}$ denote the m-vector of these variables. Let k be a fixed quantity. The constraints are represented by

$$\mathbf{A}\cdot\mathbf{x}\leq k\cdot\mathbf{u}_n \tag{4.2}$$

where $\mathbf{u}_n$ is the n-vector of all ones. Consider the 0—1 integer linear program:

$$\text{Maximize } \sum_{i=1}^{m} x_i, \quad \text{s.t. (4.2)}, \ x_i\in\{0, 1\}, \tag{4.3}$$

As usual, we solve the LP relaxation with $x_i\in[0, 1]$. Let $\tilde{v}_k^*$ be the optimum value of the objective function. Instead of directly proceeding to the randomization phase, we multiply the optimal values $\hat{x}_i$ for the variables by the quantity $1-\delta$; the computation of δ is described below. Let

$$x_i' = \hat{x}_i\cdot(1-\delta). \tag{4.4}$$

In the randomization stage, we now use the values x_i' as the probabilities rather than the $\hat{x}_i$. After rounding, the expected sum of any row of (4.2) is no more than $k\cdot(1-\delta)$. The expected value of the objective function is $\tilde{v}_k^*\cdot(1-\delta)$. In proving the quality of the rounded solution, we require a version of theorem 1.2 that deals with deviations *below* the mean of the binomial distribution:

Theorem 4.1. (Angluin and Valiant [2]). *For* $0<\beta\leq 1$,

$$P[\Psi\leq(1-\beta)Np] < \exp\left(-\frac{\beta^2 Np}{2}\right). \quad \blacksquare \tag{4.5}$$

Using theorem 4.1 together with theorems 1.1 and 1.2, we can now prove

Theorem 4.2. *Let δ_1 and δ_2 be positive constants such that $\delta_2>n\cdot e^{-k/6}$ and $\delta_1+\delta_2<1$. Let $\alpha=(3/k)\ln(n/\delta_2)$ and*

$$v_k' = \tilde{v}_k^*\cdot(1-\delta) = \tilde{v}_k^*\cdot\left(1-\frac{(a^2+4\alpha)^{1/2}-\alpha}{2}\right). \tag{4.6}$$

Then there exists an integer solution to (4.3) *satisfying*

$$\tilde{v}_k \geq v_k' - \left(2v_k'\ln\frac{1}{\delta_1}\right)^{1/2}. \tag{4.7}$$

Remark. In essence, Theorem 4.2 guarantees the existence of an integer solution of value $v_k'-O((v_k')^{1/2})$.

Proof. We first show that for the choice of δ in equation (4.6),

$$\text{(4.8)} \qquad \text{Prob. [A constraint is violated]} < \frac{\delta_2}{n}.$$

This follows directly from theorems 1.1 and 1.2, with $\beta=\delta/(1-\delta)$. The condition on δ_2 in the statement of Theorem 4.2 guarantees that $\alpha<1/2$ and thus that $\beta<1$. Thus the probability that any of the n constraints is violated is less than δ_2. By theorem 4.1,

$$\text{(4.9)} \qquad \text{Prob. [(4.7) is violated]} \leqq \delta_1.$$

Since $\delta_1+\delta_2<1$, the statement of the theorem follows.

Note that Theorem 1.2 applies only if $k>6\cdot\ln n$, for otherwise $\delta\geqq 1/2$.

For the sake of variety, we have chosen to illustrate an existence result here, rather than an algorithm as in the previous sections. By introducing a parameter ε representing the failure probability, we can modify the above theorem so that the probability that the procedure succeeds is $1-\varepsilon$ rather than merely non-zero. This provides us with a provably good algorithm for simple k-matching.

Consider the following modification of (4.2):

$$\text{(4.10)} \qquad \text{Maximize} \sum_{i=1}^{m} x_i \quad \text{s.t.} \quad \mathbf{A}\cdot\mathbf{x} \leqq \mathbf{r}$$

where $\mathbf{r}$ is an n-vector of RHS values r_i, $1\leqq i\leqq n$. This may be thought of as a resource allocation problem where r_i units of resource i are available. Each of m jobs requires one unit of each of various resources; if all resources necessary for a job are available, it can be scheduled. We wish to maximize the number of jobs scheduled.

The following Theorem is analogous to Theorem 4.2.

Theorem 4.3. *Let δ_1 and δ_2 be positive constants such that $\delta_1+\delta_2<1$. Let*

$$\text{(4.11)} \qquad v_k' = \tilde{v}_k^*(1-\delta)$$

where $\tilde{v}_k^*$ is the rational optimum of (4.10). *If there exists a constant δ in the interval* $(0,1/2]$ *such that*

$$\text{(4.12)} \qquad \sum_{i=1}^{n} \exp\left[-\frac{\delta^2 r_i}{3(1-\delta)}\right] < \delta_2$$

then there exists an integer solution to (4.10) *with objective function value at least*

$$\text{(4.13)} \qquad v_k' - \left(2v_k' \ln\frac{1}{\delta_1}\right)^{1/2}.$$

Proof. Similar to Theorem 4.2. ∎

5. Conclusions

Our results from the preceding sections deal with a class of 0—1 optimization problems. In sections 2 and 3, we developed solutions to routing and multicommodity flow problems that were close to the best possible solution. In section 4, we studied a matching problem and a resource allocation problem. In both cases, we were able to show the existence of solutions close to the rational optimum.

We have been able to apply randomized rounding only to 0—1 optimization problems with a very special structure. Furthermore, even for such structured problems, we require that the problem parameters lie in specific ranges in order that the technique be effective. For instance, in the k-matching problem in section 4, k had to be $\Omega(\log n)$. We have recently made some progress towards removing these restrictions. In joint, unpublished work with Joel Spencer, we have derived forms of the Chernoff bound that are tighter than the Angluin-Valiant bounds (theorems 1.2 and 4.1) used in this paper. The new bounds will appear in [15].

It is worth examining the tightness of our results. In general, there are two main factors that make the bounds loose. Analysis of the sum of independent Bernoulli trials of success probabilities $p_1, p_2, \ldots, p_N$ shows that Hoeffding's inequality is tightest when the probabilities are equal. If the probabilities in any problem instance Π_R span a wide range, Hoeffding's bound is weak. A second weakness of our bounds is that they relate a feasible 0—1 solution to the rational optimum, not to the 0—1 optimum. In some problem instances, the 0—1 optimum differs significantly from the rational optimum; our bounds would then be closer to the best possible than is suggested by our theorems.

Acknowledgement. We are grateful to Richard Karp for his assitance at various points in this work, and to Narendra Karmarkar for suggesting that a probabilistic technique could be used for the VLSI routing problem described in section 2. We would also like to thank the referees for their many interesting and useful suggestions and comments.

References

[1] R. Aharoni, P. Erdős and N. Linial, Dual Integer Linear Programs and the Relationship between their Optima, *Proceedings of the Seventeenth ACM Symposium on Theory of Computing, ACM, New York, May 1985,* 476—483.

[2] D. Angluin and L. G. Valiant, Fast probabilistic algorithms for Hamiltonian circuits and matchings, *Journal of Computer and System Sciences,* **19**, 155—193.

[3] H. Chernoff, A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations, *Annals of Math. Stat.,* **23** (1952), 493—509.

[4] S. Even, A. Itai and A. Shamir, On the complexity of timetable and multi-commodity flow problems, *SIAM Journal on Computing,* **5** (1976), 691—703.

[5] Z. Füredi, Maximum degree and fractional matchings in uniform hypergraphs, *Combinatorica* **1** (1981), 155—162.

[6] W. Hoeffding, On the distribution of the number of successes independent trials, *Annals of Math. Stat.,* **27** (1956), 713—721.

[7] T. C. Hu and M. T. Shing, A Decomposition Algorithm for Circuit Routing, *Mathematical Programming Study,* **24** (1985), 87—103.

[8] N. Karmarkar, A new polynomial-time algorithm for linear programming, *Combinatorica* **4** (1984), 373—396.

[9] R. M. Karp, Reducibility among combinatorial problems, *Complexity of Computer Computations,* (ed. R. N. Miller, J. W. Thatcher), *Plenum Press, New York,* (1972), 85—104.

[10] R. M. KARP, F. T. LEIGHTON, R. L. RIVEST, C. D. THOMPSON, U. VAZIRANI and V. VAZIRANI Global Wire Routing in Two-Dimensional Arrays, *Proc. 24th Annual Symp. on Foundations of Computer Science,* (1983), 453—459.
[11] L. LOVÁSZ, On the ratio of optimal and fractional covers, *Discrete Mathematics,* 13 (1975), 383—390.
[12] V. M. MALHOTRA, M. P. KUMAR and S. N. MAHESHWARI, An $O(|V|^3)$ algorithm for finding maximum flows in networks, *Information Processing Letters,* 7 (1978), 277—278.
[13] P. RAGHAVAN and C. D. THOMPSON, Randomized Routing in Gate-Arrays, *CSD/84/202, Computer Science Division, UC Berkeley,* (1984).
[14] P. RAGHAVAN and C. D. THOMPSON, Provably Good Routing in Graphs: Regular Arrays, *Proceedings of the Seventeenth ACM Symposium on Theory of Computing, ACM, New York,* (1985), 79—87.
[15] P. RAGHAVAN, Randomized Rouding and Discrete Ham-Sandwiches: Provably Good Algorithms for Routing and Packing Problems, *PhD Thesis, Computer Science Division, UC Berkeley,* (1986).

Prabhakar Raghavan

Computer Science Division
573 Evans Hall
U.C. Berkeley, CA 94720 U.S.A.

Clark D. Tompson

Computer Science Division
573 Evans Hall
U.C. Berkeley, CA 94720 U.S.A.