RC23417 (W0411-056) November 8, 2004
Computer Science

IBM Research Report

Solving Sparse Semi-Random Instances of Max Cut and Max CSP in Linear Expected Time

Alexander D Scott, Gregory B. Sorkin
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

SOLVING SPARSE SEMI-RANDOM INSTANCES OF MAX CUT AND MAX CSP IN LINEAR EXPECTED TIME

ALEXANDER D. SCOTT AND GREGORY B. SORKIN

Abstract. We show that a maximum cut of a random graph below the giant-component threshold can be found in linear space and linear expected time by a simple algorithm. In fact, the algorithm solves a more general class of problems, namely binary 2-variable-constraint satisfaction problems, or Max 2-CSPs. In addition to Max Cut, Max 2-CSPs encompass Max Dicut, Max 2-Lin, Max 2-Sat, Max-Ones-2-Sat, maximum independent set, and minimum vertex cover. We show that if a Max 2-CSP instance has an "underlying" graph which is a random graph $G(n, c/n)$, then the instance is solved in expected linear time if $c \leq 1$. Moreover, for arbitrary values (or functions) $c > 1$ an instance is solved in expected time $n \exp(O(1 + (c-1)^3/n))$; in the "scaling window" $c = 1 + \lambda n^{-1/3}$ with λ fixed, this expected time remains linear.

Our method is to show, first, that if a Max 2-CSP has a connected underlying graph with n vertices and m edges, then $nO(2^{(m-n)/2})$ is a deterministic upper bound on the solution time. Then, analyzing the tails of the distribution of this quantity for a component of a random graph yields our result. Towards this end we derive some useful properties of binomial distributions and simple random walks.

1. Introduction

In this paper we prove that a maximum cut of a random graph below the giant-component threshold can be found in linear expected time, as stated in the following theorem.

Theorem 1. *For any $c \leq 1$, a maximum cut of a random graph $G(n, c/n)$ can be found in time whose expectation is $O(n)$, using space $O(m+n)$, where m is the size of the graph.*

We should point out the difference between requiring linear time "in expectation" rather than just "almost always". With high probability, a random graph below the giant-component threshold consists solely of trees and unicyclic components, and a maximum cut in such a graph is easy to find. (It cuts all edges except for one edge in each odd cycle.) However, exponential time can be spent on finding optimal cuts in the rare multicyclic graphs, which makes the proof of Theorem 1 rather more delicate.

Our approach is to give a deterministic algorithm and an upper bound on its running time as a function of the input graph's "excess" $m - n$, where n is the order of the graph (the number of vertices) and m its size (number of edges). (Throughout, we will reserve the symbols n and m for these roles.) We then bound the expected running time for random instances by bounding the distribution of the excess for the components of a sparse random graph, as follows.

Theorem 2. *Let G be a connected graph with n vertices and m edges. A maximum cut of G can be found in time $O(m+n)2^{(m-n)/2}$, using space $O(m+n)$.*

In fact, the algorithm employs local reductions that take us outside the class of Max Cut problems, forcing us to work with the larger class Max 2-CSP: weighted maximum constraint satisfaction problems consisting of constraints on variables and pairs of variables, where each variable may take two values. This has the benefit that our results are quite a bit more general: Theorems 1 and 2 for Max Cut are just special cases of identical results for Max 2-CSP, which is the context in which we prove them.

In a forthcoming paper [SS], we use a similar approach to prove that *arbitrary* n-variable m-clause Max 2-CSP instances can be solved deterministically in time $\tilde{O}\left(2^{19m/100}\right)$. (Weaker versions of this result, and of results from the present paper, appeared in the conference paper [SS03].) This improves on previous algorithms for particular problems, notably an $\tilde{O}\left(2^{m/5}\right)$ Max 2-Sat algorithm by Gramm, Hirsch, Niedermeier and Rossmanith [GHNR03], and an $\tilde{O}\left(2^{m/4}\right)$ Max Cut algorithm by Kulikov and Fedin [KF02].

We feel that working in the larger class Max 2-CSP is a key element in obtaining both sets of results. In order to stay within a narrower domain such as Max Cut or Max 2-Sat, previous approaches needed more complicated, more numerous, and less strong reductions. The broader class Max 2-CSP allows a small number of simple and powerful reductions.

1.1. **Context.** Our results are particularly interesting in the context of phase transitions for various maximum constraint-satisfaction problems. Since we are just situating our results, we will be informal. It is well known that a random 2-Sat formula with "density" $c < 1$ (where the number of clauses is c times the number of variables) is satisfiable with probability tending to 1 as the number n of variables tends to infinity, while for $c > 1$, the probability of satisfiability tends to 0 as $n \to \infty$: see [CR92, Goe96, FdlV92] and, for more detailed results, [BBC$^+$01]. More recently, Max 2-Sat has been shown to exhibit similar behavior, so for $c < 1$, only an expected $\Theta(1/n)$ clauses go unsatisfied, while for $c > 1$, an expected $\Theta(n)$ clauses must go unsatisfied [CGHS].

For a random graph $G(n, c/n)$, with $c < 1$ the graph almost surely consists solely of small trees and unicyclic components, while for $c > 1$, it almost surely contains a "giant", complex component, of order $\Theta(n)$ [Bol01]. Again, [CGHS] proves the related facts that in a maximum cut of such a graph, for $c < 1$ only an expected $\Theta(1)$ edges fail to be cut, while for $c > 1$ it is $\Theta(n)$.

For both Max 2-Sat and Max Cut, it seems likely that the mostly-satisfiable (or mostly-cuttable) sparse instances are algorithmically easy, while the not-so-satisfiable dense instances are algorithmically hard. While, as far as we are aware, little is known about the hardness of dense instances, our results here confirm that not only are typical sparse Max Cut instances easy, but even the atypical ones can be accommodated in polynomial expected time; see the Conclusions for further discussion.

More generally, our interest here is in solving random instances of hard problems in polynomial expected time (specifically, in linear expected time), and of course there is a substantial body of literature on this subject. For example, results on

coloring random graphs in polynomial expected time can be found in [KV02, COMS, TCO03].

1.2. **Outline of proof.** Our main result will be Theorem 1, generalized from $c \leq 1$ to a larger range, and from Max Cut to the class Max 2-CSP (to be defined in Section 2). Its proof has a few principal components. Since the maximum cut of a graph is the combination of maximum cuts of each of its connected components (and the same is true for any Max 2-CSP), it suffices to bound the expected time to partition the component containing a fixed vertex.

In Theorem 5 we show that the running time of "Algorithm A" (introduced in Section 3) on a component is bounded by a function of the component's excess (the number of edges minus the number of vertices).

Lemma 17 provides a bound on the exponential moments of the excess of a component of a random graph. It does so by "exploring" the component as a branching process, dominating it with a similar process, and analyzing the latter as a random walk. This gives stochastic bounds on the component order u and, conditioned upon u, the "width" w (to be defined later); then it is easy to obtain a stochastic bound on the excess in terms of u and w.

Finally, we combine the running times, which are exponentially large in the excess, with the exponentially small large-deviation bounds on the excess, to show that Algorithm A runs in polynomial expected time.

Let us remark that we could also have tried to bound the expected running time of Algorithm A by using estimates for $C(n, n+k)$, the number of connected graphs on n vertices with $n+k$ edges. Such estimates are given in [Bol84, Łuc90, BCM90], but they seem not to be immediately suitable for our purposes; we discuss this more extensively in Section 6.

2. Max 2-CSP

The problem Max Cut is to partition the vertices of a given graph into two classes so as to maximize the number of edges "cut" by the partition. Think of each edge as being a function on the classes (or "colors") of its endpoints, with value 1 if the endpoints are of different colors, 0 if they are the same: Max Cut is equivalent to finding a 2-coloring of the vertices which maximizes the sum of these edge functions. This view naturally suggests a generalization.

An *instance* (G, S) of Max 2-CSP is given by an "underlying" graph $G = (V, E)$ and a set S of "score" functions. Writing $\{R, B\}$ for the colors Red and Blue, for each edge $e \in E$ there is a "dyadic" score function $s_e : \{R, B\}^2 \to \mathbb{R}$, for each vertex $v \in V$ there is a "monadic" score function $s_v : \{R, B\} \to \mathbb{R}$, and finally there is a single "niladic" score function $s_0 : \{R, B\}^0 \to \mathbb{R}$ which takes no arguments and is just a constant convenient for bookkeeping. We allow an instance to have parallel edges (and further such edges may be generated while the algorithm runs).

A potential *solution* is a "coloring" of the vertices, i.e., a function $\phi : V \to \{R, B\}$, and an optimum solution is one which maximizes

$$s(\phi) \doteq s_0 + \sum_{v \in V} s_v(\phi(v)) + \sum_{uv \in E} s_{uv}(\phi(u), \phi(v)). \tag{1}$$

We don't want to belabor the notation for edges, but we wish to take each edge just once, and (since s_{uv} need not be a symmetric function) with a fixed notion of which endpoint is "u" and which is "v". We will typically assume that $V = [n]$ and

any edge uv is really an ordered pair (u, v) with $1 \leq u < v \leq n$. We remark that the "2" in the name Max 2-CSP refers to the fact that the score functions take 2 or fewer arguments (3-Sat, for example, is out of scope); replacing 2 by a larger value would also mean replacing the underlying graph with a hypergraph.

An obvious computational-complexity issue is raised by allowing scores to be arbitrary *real* values. Our algorithm will add, subtract, and compare these values (never introducing a value larger than the sum of those in the input) and we assume that each such addition can be done in time $O(1)$ and represented in space $O(1)$. If desired, scores may be limited to integers, and the length of the integers factored in to the algorithm's complexity, but this seems uninteresting and we will not remark on it further.

Our assumption of an undirected underlying graph is sound even for a problem such as Max Dicut (maximum directed cut). Here one normally thinks of a directed edge as cut only if its head has color 0 and its tail has color 1, but for a directed edge (v, u) with $v > u$ this may be expressed by the undirected (or, equivalently, canonically directed) edge (u, v) with score 1 if $(\phi(u), \phi(v)) = (1, 0)$, and score 0 otherwise. That is, instead of directing the *edges* we incorporate the direction into the score functions. (In cases like this we do not mention the monadic and niladic score functions; they are "unused", i.e., taken to be identically 0.)

We can solve minimization problems by replacing each score function with its negation (there is no assumption of positivity) and solving the resulting maximization problem. Max 2-CSP also models *weighted* problems: assigning a weight to a constraint just means multiplying the score function by the weight.

Max 2-CSP also includes problems that are not obviously structured around pairwise constraints. Our original example of Max Cut may fall into this category, as do maximum independent set and minimum dominating set. To model the problem of finding a maximum independent set in a graph as a Max 2-CSP, let $\phi(v) = 1$ if vertex is to be included in the set and 0 otherwise, define vertex scores $s_v(\phi(v)) = \phi(v)$ (rewarding a vertex for being included in the set), and define edge scores $s_{uv}(\phi(u), \phi(v)) = -2$ if $\phi(u) = \phi(v) = 1$, and 0 otherwise (penalizing violations of independence, and outweighing the reward for inclusion). Similarly, for minimum dominating set we penalize vertices for inclusion, but more heavily penalize edges neither of whose endpoints is included.

3. Solving a maximum constraint-satisfaction instance

We begin by describing our algorithm, and analyze its performance on random instances in a later section.

3.1. **Algorithm A.**

In this section we give an algorithm for solving instances of weighted Max 2-CSP. The algorithm will use 3 types of reductions. We begin by defining these reductions. We then show how the algorithm fixes a sequence in which to apply the reductions by looking at the underlying graph of the instance. This sequence defines a tree of instances, which can be solved bottom-up to solve the original one. Finally, we bound the algorithm's time and space requirements.

The first two reductions each produce equivalent problems with fewer vertices, while the third produces a pair of problems, both with fewer vertices, one of which is equivalent to the original problem.

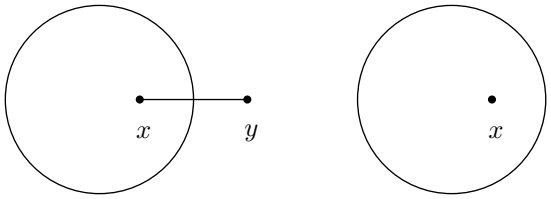
Reduction I: Let y be a vertex of degree 1, with neighbor x. Reducing (V, E, S) on y results in a new problem (V', E', S') with $V' = V \setminus y$ and $E' = E \setminus xy$. S' is the restriction of S to V' and E', except that for $C, D \in \{R, B\}$ we set

$$s'_x(C) = s_x(C) + \max_D\{s_{xy}(C, D) + s_y(D)\},$$

i.e., we set

$$s'_x(R) = s_x(R) + \max\{s_{xy}(R, R) + s_y(R), s_{xy}(R, B) + s_y(B)\}$$
$$s'_x(B) = s_x(B) + \max\{s_{xy}(B, B) + s_y(B), s_{xy}(B, R) + s_y(R)\}.$$

Note that any coloring ϕ' of V' can be extended to a coloring of V in two ways, namely ϕ_R and ϕ_B (corresponding to the two colorings of y); and the defining property of the reduction is that $S'(\phi') = \max\{S(\phi_R), S(\phi_B)\}$. In particular, $\max_{\phi'} S'(\phi') = \max_\phi S(\phi)$, and an optimal coloring ϕ' for the instance (V', E', S') can be extended to an optimal coloring ϕ for (V, E, S), in constant time.



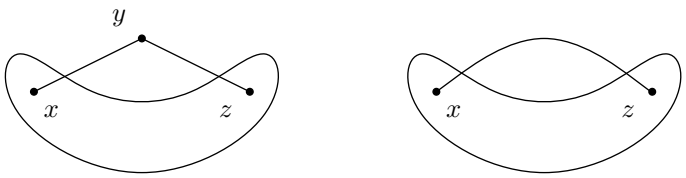
Reduction II: Let y be a vertex of degree 2, with neighbors x and z. If $x = z$ we have a pair of parallel edges: we combine the two edges and perform a type I reduction. Otherwise, reducing (V, E, S) on y results in a new problem (V', E', S') with $V' = V \setminus y$ and $E' = (E \setminus \{xy, yz\}) \cup \{xz\}$. S' is the restriction of S to V' and E', except that for $C, D, E \in \{R, B\}$ we set

$$s'_{xz}(C, D) = \max_E\{s_{xy}(C, E) + s_{yz}(E, D) + s_y(E)\},$$

i.e., we set

$$s'_{xz}(R, R) = \max\{s_{xy}(R, R) + s_{yz}(R, R) + s_y(R), s_{xy}(R, B) + s_{yz}(B, R) + s_y(B)\}$$
$$s'_{xz}(R, B) = \max\{s_{xy}(R, R) + s_{yz}(R, B) + s_y(R), s_{xy}(R, B) + s_{yz}(B, B) + s_y(B)\}$$
$$s'_{xz}(B, R) = \max\{s_{xy}(B, R) + s_{yz}(R, R) + s_y(R), s_{xy}(B, B) + s_{yz}(B, R) + s_y(B)\}$$
$$s'_{xz}(B, B) = \max\{s_{xy}(B, R) + s_{yz}(R, B) + s_y(R), s_{xy}(B, B) + s_{yz}(B, B) + s_y(B)\}.$$

This reduction creates a new edge xy, which may be parallel to one or more existing such edges, each such edge having its associated score function. (Unfortunately our notation fails to distinguish the various edges and scores, but this is only to keep the notation manageable; there is no deeper issue.) As in Reduction I, any coloring ϕ' of V' can be extended to V in two ways, ϕ_R and ϕ_B, and S' picks out the larger of the two scores. Also as in Reduction I, $\max_{\phi'} S'(\phi') = \max_\phi S(\phi)$, and an optimal coloring ϕ' for the instance (V', E', S') can be extended to an optimal coloring ϕ for (V, E, S), in constant time.



Reduction III: Let y be a vertex of degree 3 or higher. Where reductions I and II each had a single reduction of (V, E, S) to (V', E', S'), here we define a pair of reductions of (V, E, R), to (V', E', S^R) and (V', E', S^B), corresponding to assigning the color R or B to y. We define $V' = V \setminus y$, and E' as the restriction of E to $V \setminus y$. For $C, D, E \in \{R, B\}$, S^C is the restriction of S to $V \setminus y$, except that we set

$$(s^C)_0 = s_0 + s_y(C),$$

and, for every neighbor x of y,

$$(s^C)_x(D) = s_x(D) + s_{xy}(D, E).$$

In other words, S^R is the restriction of S to $V \setminus y$, except that we set $(s_0^C) = s_0 + s_y(C)$ and, for every neighbor x of y,

$$(s^R)_x(R) = s_x(R) + s_{xy}(R, R) + s_y(R)$$
$$(s^R)_x(B) = s_x(B) + s_{xy}(B, R) + s_y(R).$$

Similarly S^B is given by $(s^B)_0 = s_0 + s_y(B)$ and, for every neighbor x of y,

$$(s^B)_x(R) = s_x(R) + s_{xy}(R, B) + s_y(B)$$
$$(s^B)_x(B) = s_x(B) + s_{xy}(B, B) + s_y(B).$$

As in the previous reductions, any coloring ϕ' of $V \setminus y$ can be extended to V in two ways, ϕ_R and ϕ_B, corresponding to the color given to y, and now (this is different!) $S_R(\phi') = S(\phi_R)$ and $S_B(\phi') = S(\phi_B)$. Furthermore,

$$\max\{\max_{\phi'} S_R(\phi'), \max_{\phi'} S_B(\phi')\} = \max_{\phi} S(\phi),$$

and an optimal coloring on the left can be extended to an optimal coloring on the right in time $O(\deg(y))$.

Reduction 0: We define one more "pseudo-reduction". If a vertex y has degree 0 (so it has no dyadic constraints), for $C \in \{R, B\}$ we set

$$s_0 = s_0 + \max_C s_y(C)$$

and we delete y from the instance entirely.

3.2. **Algorithm idea.** The natural algorithm for solving an input instance would work as follows. Begin with the input problem instance. Given an instance $\mathcal{M} = (G, S)$, if any reduction of type 0, I or II is possible, apply it to reduce $\mathcal{M}$ to $\mathcal{M}'$, record certain information about the reduction, solve $\mathcal{M}'$ recursively, and use the recorded information to extend the solution to one for $\mathcal{M}$. If only a type III reduction is possible, first recursively solve $\mathcal{M}^R$ (the "red" version of the reduction), then solve $\mathcal{M}^B$ (the "blue" version), select the solution with the larger score, and use the recorded information to extend the solution to one for $\mathcal{M}$. If no reduction is possible then the graph has no vertices, the coloring solution is a length-0 vector, and the score is s_0.

That the algorithm returns an optimal solution, i.e., that it is correct, follows from the definitions of the reductions. The recursive computation implicitly defines a computation tree. If from the root to any leaf in this tree the number of type III reductions is at most r, then a natural implementation (storing the left-child solution as the right child is solved) would require space $O(rn)$. We now describe a refinement which as we will show uses space only $O(n)$ and (like the simple recursive algorithm) takes time $O(n2^r)$.

3.3. **Algorithm implementation and analysis.** As we shall show in Section 3.4, each type 0, I, or II reduction from an instance $\mathcal{M}$ to $\mathcal{M}'$ can be identified and performed in time and space $O(1)$, and "annotated" in time and space $O(1)$ so that from the instance $\mathcal{M}'$, its optimal solution, and the annotation, we can in additional time and space $O(1)$ reconstruct the instance $\mathcal{M}$ and construct its optimal solution. Similarly, each type III reduction from an instance $\mathcal{M}$ to an instance $\mathcal{M}^R$ (respectively $\mathcal{M}^B$) can be identified in time and space $O(1)$, performed in time and space $O(n)$, and recorded in time $O(n)$ by an $O(n)$-space annotation so that from the "red" instance $\mathcal{M}^R$ (respectively the blue instance $\mathcal{M}^B$), its optimal solution, and the annotation, we can in time $O(1)$ reconstruct the instance $\mathcal{M}$ and construct a solution which is optimal *among* solutions with the reduction vertex colored red (respectively blue).

From the reduction tree, define a "reduced tree" in which each maximal path consisting only of type 0, I and II reductions is compressed into a single edge. Since each edge in the path can be found, annotated, and traversed in either direction using time and space $O(1)$, each compressed edge can be found, annotated, and traversed in either direction using time and space $O(n)$. All non-leaf vertices in this reduced computation tree have two children, and the maximum depth r of the reduced tree is the maximum number of type III reductions along any root-to-leaf path in the original tree. (In fact both trees have all their leaves at equal depth, but this is immaterial for us.) To find the optimum solution in this tree can be expressed in a simpler abstract form. (in which we replace the $O(n)$ time and space costs with $O(1)$ for convenience).

Given a rooted tree T, having values associated with its leaves, we wish to return the location of a maximum-value leaf. The tree is given in the following implicit form. We begin at the root, which so identifies itself. In unit time and space we can do all the following: from each non-leaf node, and for any integer $k > 0$, navigate down to the kth child (or be informed if it has fewer than k children); from each non-root vertex, navigate up to its parent; at each leaf vertex, find the value. The solution is to be returned as a path from the root to an optimum leaf, that is, a

string of numbers $k_1, k_2, \ldots$ signifying that a maximum-value leaf is to be found as child k_1 of the root, child k_2 of that, etcetera.

Claim 3. *A path from the root to an optimum leaf of an n-node tree T of maximum depth r, specified implicitly as described above, can be computed by a suitable implementation of depth-first search in time $O(n)$ and space $O(r)$.*

Proof. The depth-first search algorithm maintains a value $v^\star$, depth $d^\star$, and location string $\{k_i^\star\}_{i=1}^{d^\star}$ for the best leaf seen so far (initially $v^\star = -\infty$, $d^\star = 0$, and $k^\star = \emptyset$); a depth d (initially 0) and string $\{k_i\}_{i=1}^{d}$ indicating the current vertex; a counter c (initially 0) for the current vertex's last-visited child; and another counter j (initially 0) indicating the depth at which the root-to-vertex paths to the best-so-far leaf and the current vertex first diverge (so that $k_i^\star$ and k_i are equal for all $i < j$ and unequal for $i = j$). These data structures are clearly of size only $O(r)$, the maximum depth of the tree. As depth-first search proceeds, if a new record leaf is found at depth d, we update the best-so-far records by setting $v^\star$ equal to the leaf's value v, $d^\star$ to the leaf's depth d, and $\{k_i^\star\}_{i=j}^{d} = \{k_i\}_{i=j}^{d}$.

At each up or down step of depth-first search, $\{k\}_i$, c, and d can be updated in time $O(1)$. So can j, which is normally updated to the smaller of its old value and the current node depth d, or set to the leaf depth d when a new record is achieved. To show that the total number of changes to $k^\star$ is $O(n)$ rather than the naive bound of $O(n^2)$, first note that when changing from a record leaf $l^\star$ to a new record leaf l, the number of updates to $k^\star$ is less than the path distance $\mathrm{dist}(l^\star, l)$ from $l^\star$ to l. It follows that the total number of updates to $k^\star$ is at most the depth of the first record leaf plus the sum over records l_i of the inter-pair distances, $\sum_i \mathrm{dist}(l_{i+1}, l_i)$. Because this distance obeys the triangle inequality, defining a first pseudo record leaf l_0 with depth 0, if there are L leaves in all, the total number of updates is at most $\sum_{i=0}^{L-1} \mathrm{dist}(l_i, l_{i+1})$. The paths underlying these distances precisely comprise a depth-first search tour of the tree, so that the above sum is precisely twice the number of edges of the tree, or $2(n-1)$ for an n-node tree. □

Corollary 4. *An n-vertex, m-constraint Max 2-CSP instance whose computation tree has at most r type III reductions in any path can be solved in time $O(n2^r)$ using space $O(n+m)$.*

Proof. Since the reduced Max 2-CSP search tree is binary, its size is 2^r. By the lemma, with $O(1)$-time operations the solution to the abstracted tree problem can be found in time $O(2^r)$, which for the actual $O(n)$-time reductions translates into time $O(n2^r)$. Solving the abstract tree problem takes space only $O(r)$, which is subsumed by the $O(m)$ space we now show is sufficient to maintain a root-to-node path in the CSP instance, which is all that is required. Each reduction of type 0, I and II affects at most 3 vertices and at most 3 edge constraints, and thus can be recorded in constant space; since each removes at least one vertex there are at most n of them. Each type III reduction on a vertex of degree d affects $d+1$ vertices and d edge constraints, and thus can be recorded in space $O(d)$; since the degrees never increase above their initial values, the total space for all type III reductions is at most the sum of all the initial vertex degrees, or $2m$. □

We can now bound the running time of Algorithm A in terms of the excess of the graph underlying the CSP.

Theorem 5. *Given a weighted Max 2-CSP whose underlying graph G is connected, has order n, size m, and excess $\kappa = m - n$, Algorithm A returns an optimal solution in time $O(n)2^{\kappa/2}$.*

Proof. In light of Corollary 4, it suffices to prove that the number of type-III reduction steps $r(G)$ is bounded by $\max\{0, \kappa/2\}$.

The proof is by induction on the order of G. If G has excess 0 (it is unicyclic) or excess -1 (it is a tree), then type-I and -II reductions destroy all its edges, so $r = 0$.

Otherwise, the first type-III reduction, from G to G', reduces the number of edges by at least 3 and the number of vertices by exactly 1, thus reducing the excess to $\kappa' \leq \kappa - 2$. If G' has components $G'_1, \ldots, G'_I$, then $r(G) = 1 + \sum_i r(G'_i)$. Given that we applied a type-III reduction, G had minimum degree ≥ 3, so G' has minimum degree ≥ 2. Thus each component G'_i has minimum degree ≥ 2, and so excess $\kappa'_i \geq 0$. Then, by induction, $r(G) = 1 + \sum_i r(G'_i) \leq 1 + \sum_i \max\{0, \kappa'_i/2\} = 1 + \sum_i \kappa'_i/2 \leq 1 + \kappa'/2 \leq \kappa/2$. □

3.4. **Implementation details and data structures.** It remains only to show that the space required for each computation-tree node is $O(1)$, and that the time is $O(1)$ for each type 0, I and II reduction, and $O(d)$ for a type III reduction on a vertex of degree d. Both of these depend on implementation details unrelated to the main direction of the paper, the characterization of a random process. However, since the linear-time result depends on these tight space and time bounds, we now sketch out an efficient implementation.

3.4.1. *Data structure.* We assume a RAM model, so that a given memory location can be accessed in constant time.

We presume that the input graph is given in a sparse representation, consisting of a vector of nodes, and with each node: its monadic score function (a 2-element table) and a linked list of incident edges, each edge with its dyadic score function (a 4-element table) and a pointer to the edge's twin copy indexed by the other endpoint. We also maintain the degree of each vertex; an indication of whether it is still unset or has been set to Red or Blue; and a stack containing all vertices of degrees 0, 1, and 2; these can all be created in linear time from the input. Starting with a doubly linked list of all the vertices, as vertices are removed from an instance to create a subinstance, they are also bridged over in the linked list, so that there is always a linked list of just the vertices in the subinstance.

3.4.2. *Transformations.* If the stack is non-empty, we pop the first vertex off it and perform a type 0, I or II reduction on it. We omit a discussion of type 0 and I reductions and start with the slightly more complicated type-II reductions. Suppose that the popped vertex y has just two neighbors, x and z. First we construct the score function $s_{xz}(,)$ replacing $s_{xy}(,)$ and $s_{yz}(,)$, per the reduction step above. At the same time, we make a note of how to set y as a function of x and z. For example if xy is a "cut" constraint of weight 2, and yz is a cut constraint of weight 1, these are replaced by a single anti-cut constraint on xz, with associated optimal values of y: in figure 1 the first table gives the score function for xy, the second gives that for yz, and the third, "table T" gives the score function and the optimal value of y for xz. Table T, mapping the coloring of xz to a score and an optimal color for y, is associated with the instance $\mathcal{M}$ being reduced. Then, the edge and score

y	x	score
R	R	0
R	B	2
B	R	2
B	B	0

y	z	score
R	R	0
R	B	1
B	R	1
B	B	0

x	z	y	score
R	R	B	3
R	B	B	2
B	R	R	2
B	B	R	3

FIGURE 1. Example of a II-reduction replacing score functions for yx and yz with a score function for xz and the associated optimal values of y.

yx are deleted by bridging over it in the doubly linked list of y's edges, while the pointer from yx is used to locate its twin xy which is replaced by the edge xz and its freshly computed score function. Similarly, edge yz is deleted, and its twin zy replaced by zx. The degree of y becomes 0, while those of x and z are unchanged. This defines the new instance $\mathcal{M}'$, and all takes constant time and space.

After $\mathcal{M}'$ has been solved recursively, so that x and z have been colored, y is colored according to table T, giving an optimal solution to instance $\mathcal{M}$ in constant time.

3.4.3. *Splittings.* If on the other hand the stack is empty, we must III-reduce on some vertex. We traverse the linked list of the n vertices in the subinstance to find one of degree $d \geq 3$. (If there is none then the instance has no vertices, and we return the empty coloring and the niladic score s_0.)

Once an appropriate vertex y is located, a "Red" III-reduction on y is performed by making a first sweep through the incident edges, for edge yx adding the score function $s_{yx}(,)|_{y=R}$ to the monadic score function $s_x()$, then making a second sweep and deleting each edge yx and twin xy. (As each edge is deleted, we save a pointer to it and its predecessor and successor; we also save a pointer to each neighbor x of y.) When edge xy is deleted, the degree of x is decremented; if the degree becomes 2, x is pushed onto the stack. This defines an instance $\mathcal{M}^R$, which is then solved recursively; we record the solution cost and the colors of all n vertices in the subinstance. We then reconstruct the original instance $\mathcal{M}$, using the saved pointers to "reconstruct" the deleted edges ("un-bridging" the pointer bridges we built around them, correcting the vertex degrees, and undoing the changes to the monadic score functions). The same procedure is of course applied for the reduction to $\mathcal{M}^B$.

4. The binomial distribution and binomial random walks

Our analysis in the next section will center on characterizing the order and excess of a component of a random graph, which we will do by showing how these quantities are dominated by parameters of a random walk.

Characterization of the random walk itself requires a certain amount of work, and since it is independent of our Max CSP context and likely to be of more general interest, we take it up in this separate section.

We would have expected the facts given in this section already to be known, but we have searched the literature and spoken to colleagues without turning up anything. We did not find any reference to the natural continuous extension of the binomial density function (Definition 8), that a binomial's right tail dominates

its left tail (Theorem 10 and Corollary 11), nor our true target, an exponential tail bound on the deviation of a simple random walk above its linear interpolate (Theorem 12). Even if these results are already known, they are apparently in a state of obscurity in our field at least, and given how fundamental they seem, we are glad for the chance to bring them to attention.

Our aim parallels well-known results for Brownian motion. Since we took both that result and its proof as our model, let us state it. Let $X : [0,1] \to \mathbb{R}$ be a standard Brownian motion with $X(0) = 0$ and $X(1) = s$. Then, where ϕ denotes the density of the standard normal $N(0,1)$, the following theorem is a classical result on the "standard Brownian bridge" $X(t) - ts$.

Theorem 6. *For any $b \geq 0$, $\Pr\left(\max_t(X(t) - ts) \geq b\right) = \phi(2b)/\phi(0) = \exp(-2b^2)$.*

For a standard Brownian motion, an increment $X(t+\tau) - X(t)$ has Gaussian distribution $N(0,\tau)$, and the proof of the theorem applies the reflection principle to Brownian motion, using the symmetry $\phi(x) = \phi(-x)$ of the Gaussian density).

We require an analogous result for a simple random walk $X(t)$, by which we mean a walk which at each step increases by 1 with probability p, and stays the same with probability $1-p$; we will condition the walk on $X(0) = 0$ and $X(n) = s$. The increments $X(t+\tau) - X(t)$ for the (unconditioned) random walk have binomial distribution $B(\tau, p)$, and our proof of Theorem 12 (analogous to the Brownian-motion theorem above) applies the reflection principle to the random walk, using the *asymmetry* of the binomial distribution as in Theorem 10 and Corollary 11.

We begin by defining and characterizing a continuous extension of the binomial density function.

Definition 7. *For any real values $n > 0$, $0 \leq p \leq 1$, and any k, we define*

$$B_{n,p}(k) \doteq \frac{\Gamma(n+1)}{\Gamma(k+1)\Gamma(n-k+1)} p^k (1-p)^{n-k} \tag{2}$$

if $0 \leq k \leq n$, and $B_{n,p}(k) \doteq 0$ otherwise.

For integers n and $k \in \{0, \ldots, n\}$ this is of course just the binomial density $\binom{n}{k} p^k (1-p)^{n-k}$. Although the continuous extension need not integrate to 1, we will still call it the "continuous binomial density".

It is clear that the usual binomial density function (on integers) is unimodal with with a unique maximum lying at $k = \lfloor (n+1)p \rfloor$, or two maxima if, for that k, $B_{n,p}(k) = B_{n,p}(k+1)$. We first prove that the continuous extension is unimodal.

Theorem 8. *The continuous binomial density defined by (2) is unimodal; $B_{n,p}(Np - 1) = B_{n,p}(Np)$; every value of $B_{n,p}$ on the interval $[Np-1, Np]$ exceeds every value outside it; and thus the maximum lies in this interval.*

Note that the maximum need not occur at np, as shown for instance by $n = 3$, $p = 1/3$, where the maximum occurs at around 0.82 rather than 1.

Proof. We use Gauss's representation $\Gamma(x) = x^{-1}\prod_{i=1}^{\infty}\left[(1+1/i)^x\ (1+x/i)^{-1}\right]$ [GKKH77, p. 450]. First, $B_{n,p}(k)$ is log-concave:

$$\begin{aligned}\ln B_{n,p}(k) &= k\ln p + (n-k)\ln(1-p) + \ln(k+1) + \ln(n-k+1) + \ln\Gamma(n+1)\\ &\quad - \sum_{i\geq 1}[(k+1)\ln(1+1/i) - \ln(1+(k+1)/i)]\\ &\quad - \sum_{i\geq 1}[(n-k-1)\ln(1+1/i) - \ln(1+(n-k+1)/i)],\end{aligned}$$

(3)

$$\begin{aligned}\frac{d}{dk}\ln B_{n,p}(k) &= \ln p - \ln(1-p)\\ &\quad + \sum_{i\geq 0}\left[\frac{1}{i+k+1} - \frac{1}{i+n-k+1}\right]\ \text{, and}\end{aligned}$$

$$\frac{d^2}{dk^2}\ln B_{n,p}(k) = 0 + \sum_{i\geq 0}[-1/(i+k+1)^2 - 1/(i+n-k+1)^2] < 0.$$

Thus $B_{n,p}(k)$ has a unique maximum. Also,

$$B_{n,p}(k)/B_{n,p}(k-1) = \frac{n-k+1}{k}\frac{p}{1-p}, \tag{4}$$

so for $k=(n+1)p$, $B_{n,p}(k)/B_{n,p}(k-1)=1$. Thus the maximum of $B_{n,p}(k)$ occurs for some k in the range $[(n+1)p-1,(n+1)p]$, and moreover every value of $B_{n,p}(k)$ in this range is at least as large as every value outside it. □

We will need the following simple fact.

Remark 9. *If a real-valued function f is convex on $[a-\lambda, b+\lambda]$, with $a<b$ and $\lambda\geq 0$, then*

$$\frac{1}{b-a}\int_a^b f(x)dx \leq \frac{f(a-\lambda)+f(b+\lambda)}{2}.$$

Proof. Let $\bar{f}(x)$ be the linear interpolation at x from $f(a-\lambda)$ and $f(b+\lambda)$. By convexity, for all $x\in[a-\lambda,b+\lambda]$, $f(x)\leq\bar{f}(x)$. Integrating, $\frac{1}{b-a}\int_a^b f(x)dx \leq \frac{1}{b-a}\int_a^b \bar{f}(x)dx$. As the average value of a linear function, the latter quantity is $\frac{1}{2}\bar{f}(a+b) = \frac{1}{2}(\bar{f}([a+\lambda]) + \bar{f}([b-\lambda])) = \frac{1}{2}(f(a-\lambda)+f(b+\lambda))$, concluding the proof. □

For a Gaussian distribution, the right and left tails are of course symmetric to one another. For large n and fixed p, a binomial distribution $B_{n,p}$ is approximately Gaussian and the two tails will be nearly but not exactly symmetric. We use Claim 8 to show that, for $p\leq 1/2$, a binomial's right tail (slightly) dominates its left tail. (For $p>1/2$ the opposite is true, by symmetry.)

Theorem 10. *For $p\in(0,1/2)$, the continuous binomial density function $B_{n,p}(k)$ defined by* (2) *has the property that for all deviations $\delta\geq 0$,*

$$B_{n,p}((n+1)p-1-\delta) \leq B_{n,p}([(n+1)p]+\delta).$$

Proof. For notational convenience, let $N = n + 1$. Truth of the theorem for $\delta = 0$ is immediate from Claim 8's assertion that $B_{n,p}(Np - 1) = B_{n,p}(Np)$. It suffices, then, to prove the non-negativity of

$$\frac{d}{d\delta} \ln \left(\frac{B_{n,p}(Np + \delta)}{B_{n,p}(Np - \delta - 1)} \right) = \frac{d}{d\delta} \ln B_{n,p}(Np + \delta) + \frac{d}{d\delta} \ln B_{n,p}(Np - \delta - 1).$$

(That is, the positive slope at the point $Np - \delta - 1$ should "outweigh" the negative slope at $Np + \delta$.) Taking the derivatives from (3), then, we wish to show non-negativity of

$$2(\ln(p) - \ln(1-p)) + \sum_{i \geq 0} \left[\frac{1}{i + Np + \delta + 1} - \frac{1}{i + N(1-p) - \delta} + \frac{1}{i + Np - \delta} - \frac{1}{i + N(1-p) + \delta + 1} \right]. \tag{5}$$

Before proving this, we note that each of the four parts of the summation is a harmonic sum, so for example the first sum may be approximated as $\ln(Np+\delta+1)$. Disregarding the $\delta + 1$ as being relatively small, and doing the same for the other terms, would approximate the four sums by $-2(\ln(p)-\ln(1-p))$, and expression (5) would be approximated as 0. That is, to a reasonably first-order approximation, expression (5) is 0, and it is delicate to prove that it is non-negative.

Let $f(x) = 1/[(p+x)(1-p+x)]$. Note that $(1-2p)\int_0^\infty f(x)dx = \ln(1-p)-\ln(p)$, so f will be used to address the first summand in (5). Also $f''(x) = 2/[(1-p+x)(p+x)^3] + 2/[(1-p+x)^2(p+x)^2] + 2/[(1-p+x)^3(p+x)^2]$, which is positive for $x > -p$, so f is convex on $(-p, \infty)$. Returning to the quantities in (5), then

$$\begin{aligned}
&\sum_{i \geq 0} \left[\frac{1}{i + Np + \delta + 1} - \frac{1}{i + N(1-p) - \delta} + \frac{1}{i + Np - \delta} - \frac{1}{i + N(1-p) + \delta + 1} \right] \\
&= \sum_{i \geq 0} \left[\frac{N(1-2p)}{(i + Np - \delta)(i + N(1-p) - \delta)} + \frac{N(1-2p)}{(i + Np + \delta + 1)(i + N(1-p) + \delta + 1)} \right] \\
&= \sum_{i \geq 0} \left[\frac{\frac{1}{N}(1-2p)}{(p + (i-\delta)/N)(1 - p + (i-\delta)/N)} + \frac{\frac{1}{N}(1-2p)}{(p + (i+\delta+1)/N)((1-p) + (i+\delta+1)/N)} \right] \\
&= \frac{1}{N}(1-2p) \sum_{i \geq 0} [f((1-\delta)/N) + f((1+\delta+1)/N)] \\
&\geq (1-2p) \sum_{i \geq 0} 2 \int_{i/N}^{(i+1)/N} f(x)dx \quad \text{(as explained below)} \\
&= 2(1-2p) \int_0^\infty f(x)dx \\
&= -2(\ln(p) - \ln(1-p)),
\end{aligned}$$

proving (5). The inequality follows from Remark 9, with $a = 1/N$, $b = (i+1)/N$, and $\lambda = \delta/N$: f is convex on $(-p, \infty)$, which contains the relevant range because $a - \lambda = (i - \delta)/N \geq -\delta/N > -p$ as long as $\delta \leq np < Np$, while if $\delta > np$ then the theorem is true trivially, as $B_{n,p}((n+1)p - d)$ is 0 while $B_{n,p}((n+1)p + d)$ is positive.

□

We note that this has the following corollary for binomial random variables.

Corollary 11. *For a binomially distributed random variable $X \sim B(n,p)$, $p \le 1/2$, for any $\delta \ge 0$, $\Pr(X = \lfloor (n+1)p - 1 - \delta \rfloor) \le \Pr(X = \lfloor (n+1)p + \delta \rfloor)$.*

Proof. From Claim 8, $\Pr(X = \lfloor (n+1)p - 1 - \delta \rfloor) = B_{n,p}(\lfloor (n+1)p - 1 - \delta \rfloor) \le B_{n,p}(X = (n+1)p - 1 - \delta)$. Also, $\Pr(X = \lfloor (n+1)p + \delta \rfloor) = B_{n,p}(\lfloor (n+1)p + \delta \rfloor) \ge B_{n,p}(X = (n+1)p + \delta)$: if $\delta > 1$ the last inequality follows from the fact that $B_{n,p}$ decreases above $(n+1)p$, while if $\delta < 1$ it follows from the fact that every value of $B_{n,p}$ in the interval $[(n+1)p - 1, (n+1)p]$ is larger than any value outside it. By Theorem 10, $B_{n,p}(X = (n+1)p - 1 - \delta) \le B_{n,p}(X = (n+1)p + \delta)$. Putting the three inequalities together proves the claim. □

Next we consider the deviation of a random walk above its linear interpolation. Our bound on the tail of this parameter is roughly the square of what would be obtained from a naive application of Hoeffding's inequality for sampling with replacement.

As noted earlier, our result and proof are modeled on a classical equality (Theorem 6) for the Brownian bridge. Since the "long-run" behavior of a simple random walk converges to Brownian motion (in a sense we do not need to make precise), it is not surprising that we should be able to obtain a similar result.

The probability bound obtained is roughly the square of what would be obtained from a naive application of Hoeffding's inequality for sampling with replacement [Hoe63, Section 6].

Theorem 12. *Fix any positive integers n and $S \le n/2$, and any integer discrepancy $b \ge 2$. Let $X_1, \ldots, X_n$ be a random 0-1 sequence with sum $S = X_1 + \cdots + X_n$. Then*

$$\Pr\left(\max_i \left\{ X_1 + \cdots + X_i - \frac{i}{n} S \right\} \ge b\right) \le \frac{B_{n,S/n}(S + 2b - 1)}{B_{n,S/n}(S)}.$$

Proof. First observe that a random 0-1 sequence $X_1 + \cdots + X_n = S$ as above has precisely the same distribution as a sequence of n i.i.d. Bernoulli random variables conditioned on having sum S, and in particular as such a sequence where each random variable has distribution $X_i \sim B(p)$ with $p = S/n \le 1/2$. For notational convenience, let $X^\tau = \sum_{i=1}^{\tau} X_i$. Noting that $\mathbb{E}X^\tau = \tau p$, then, we are asking the probability that there is a time τ such that $X^\tau \ge \tau p + b$. If so, define the "first crossing time" τ_b to be $\min\{\tau \le n : X^\tau \ge \tau p + b\}$, and otherwise let $\tau_b = n + 1$. Because X increases by at most 1 in a step, if $\tau_b \le n$ then $X^{\tau_b} = \lceil \tau p + b \rceil$. The probability we are interested in is precisely that $\tau_b \le n$, conditioned on $X^n = np$:

$$\Pr(\tau_b \le n \mid X^n = np) = \frac{\Pr(\tau_b \le n, X^n = np)}{\Pr(X^n = np)}. \tag{6}$$

The numerator of this expression is

$$\begin{aligned} \text{num} &= \sum_{\tau=1}^{n} \Pr(\tau_b = \tau, X^n = np) \\ &= \sum_{\tau=1}^{n} \Pr(\tau_b = \tau) \Pr(X^n = np \mid X^\tau = \lceil \tau p + b \rceil) \end{aligned}$$

$$= \sum_{\tau=1}^{n} \Pr(\tau_b = \tau) B_{n-\tau,p}(np - \lceil \tau p + b \rceil)$$
$$= \sum_{\tau=1}^{n} \Pr(\tau_b = \tau) B_{n-\tau,p}((n - \tau + 1)p - 1 - \delta)$$

where, using b's integrality, $\delta = ((n-\tau+1)p-1) - (np - \lceil \tau p + b \rceil) = b - 1 + \lceil \tau p \rceil - \tau p + p \geq 0$, Sand thus we may apply the inequality of Theorem 10:

$$\leq \sum_{\tau=1}^{n} \Pr(\tau_b = \tau) B_{n-\tau,p}((n - \tau + 1)p + \delta)$$
$$= \sum_{\tau=1}^{n} \Pr(\tau_b = \tau) B_{n-\tau,p}(np + b - 1 - 2\tau p + \lceil \tau p \rceil + 2p).$$

For reasons that will shortly become clear, we wish to replace the binomial's argument by $np + 2b - 1 - \lceil \tau p + b \rceil$. We observe that the original argument is larger than $(n-\tau+1)p$ (because $\delta \geq 0$); the new argument is smaller than the original one (because b is integral, and $-\lceil \tau p \rceil \geq -2\tau p + \lceil \tau p \rceil$); and the new argument is larger than $(n - \tau + 1)p - 1$ (because the difference is $b - \lceil \tau p \rceil + \tau p - p > b - 1 - p > 0$). Thus by Theorem 8, the binomial's value can only increase:

$$\leq \sum_{\tau=1}^{n} \Pr(\tau_b = \tau) B_{n-\tau,p}(np + 2b - 1 - \lceil \tau p + b \rceil)$$
$$= \sum_{\tau=1}^{n} \Pr(\tau_b = \tau) \Pr(X^n = np + 2b - 1 \mid X^\tau = \lceil \tau p + b \rceil)$$
$$= \sum_{\tau=1}^{n} \Pr(\tau_b = \tau, X^n = np + 2b - 1)$$
$$= \Pr(X^n = np + 2b - 1),$$

where the last equality holds because $X^n = np + 2b - 1$ means that X^n exceeds its expectation np by $2b - 1 \geq b$ and thus implies that $\tau_b \leq n$. Returning to (6) and substituting $S = np$ yields the claim. □

Remark 13. *For positive integers b and np with $b \leq 2np$,*

$$\frac{B(n, p; np + b)}{B(n, p; np)} \leq \exp\left(-(3\ln(3) - 2)/4 \ \cdot b^2/np\right).$$

Proof. The proof is by simple calculation.

$$\begin{aligned}\frac{B(n,p;np+b)}{B(n,p;np)} &= \frac{(n-(np+b))\cdots(n-np)}{(np+b)\cdots(np)}\left(\frac{p}{1-p}\right)^b \\ &= \prod_{i\le b}\frac{1-i/n(1-p)}{1+i/np} \\ &\le \prod_{i\le b}\frac{1}{1+i/np} \\ &\le \exp\left(-\int_0^b \ln(1+i/np)di\right) \\ &= \exp\left(-np(1+b/np)\ln(1+b/np)+b\right). \qquad (7)\end{aligned}$$

If $b = x\cdot np$, then the value of c for which (7) *equals* $\exp\left(-cb^2/np\right)$ is $c = \frac{(x+1)\ln(x+1)-x}{x^2}$. Since this is decreasing in x, the worst-case (smallest) value of c occurs for the largest allowed value of $x = b/np$. By hypothesis, this is $x = 2$, where $c = (3\ln(3)-2)/4$. For smaller values of $x = b/np$, then, (7) is smaller than $\exp\left(-(3\ln(3)-2)\ \cdot b^2/np\right)$, completing the proof. □

5. Stochastic size and excess of a random graph

We stochastically bound the excess $\kappa = m - n$ of a component of a random graph G via the branching-process approach pioneered by Karp [Kar90]. Given a graph G and a vertex x_1 in G, together with a linear order on the vertices of G, the branching process finds a spanning tree of the component G_1 of G that contains x_1 and, in addition, counts the number of non-tree edges of G_1 (i.e., calculates the excess minus 1).

At each step of the process, vertices are classified as "living", "dead", or "unexplored", beginning with just x_1 living, and all other vertices unexplored. At the ith step, the process takes the earliest living vertex x_i. All edges from x_i to unexplored vertices are added to the spanning tree, and the number of non-tree edges is increased by 1 for each edge from x_i to a living vertex. Unexplored vertices adjacent to x_i are then reclassified as living, and x_i is made dead. The process terminates when there are no living vertices.

Now suppose G is a random graph in $\mathcal{G}(n,c/n)$, with the vertices ordered at random. Let $w(i)$ be the number of live vertices at the ith step and define the *width* $w = \max w(i)$. (Note that $w(i)$ and w are functions of the random process, not just the component, since they depend on the order in which the vertices are taken. Despite this, for convenience we will refer to the "width of a component".)

Let $u = |G_1|$, so that $w(0) = 1$ and $w(u) = 0$. The number of non-tree edges uncovered in the ith step is binomially distributed as $B(w(i)-1, c/n)$, and so, conditioning on u and $w(1),\ldots,w(u)$, the number of excess edges is distributed as

$$B\Big(\sum_{i=1}^{u}(w(i)-1), c/n\Big). \qquad (8)$$

Since $\sum_{i=1}^{u}(w(i)-1) \le uw$, and the number of excess edges is always at most $\binom{u}{2}$, the number of excess edges is dominated by the random variable $B(\min\{uw, \binom{u}{2}\}, c/n)$. At the ith stage of the process, there are at most $n-i$ unexplored vertices, and so

the number of new live vertices is dominated by $B(n-i, c/n)$. This allows us to define a simpler random walk which dominates the graph edge-exposure branching process.

Definition 14. *Given a constant $c > 0$ and integer $n > 0$, define the random walk RW by $X(1) = 1$ and $X(i) = X(i-1) + B(n-i, c/n)$. Parametrize its time-$i$ width by $W'(i) = X(i) - i$, its width by $W' = \max_i W'(i)$, and its order by $U' = \min\{n, \min\{i \colon W'(i) = 0\}\}$.*

Claim 15. *The order U and width W of the component G_1 on a vertex 1 of a random graph $G(n, c/n)$ are stochastically dominated by U' and W' of the random walk RW.*

Proof. Consider a variant of the branching process on the random graph in which at each step we add enough new special "red" vertices to bring the number of unexplored vertices to $n - i$. This is equivalent to the random walk RW. It also dominates the original branching process: in the implicit coupling between the two, the variant has width at least as large at every step, and thus also has maximum width and order which are at least as large as those of the original process. □

Thus the excess κ_1 of G_1 is stochastically dominated by the same quantity for RW:

$$\kappa_1 \preceq B\left(\min\left\{ U'W', \binom{U'}{2} \right\}, c/n \right). \tag{9}$$

Let $t(G_1)$ be the time spent by Algorithm A on G_1. We shall analyze the total running time on by "charging" $t(G_1)/|G_1|$ to each vertex of G_1: the running time is then the sum of these charges.

Claim 16. *The amortized running time $t(G_1)/|G_1|$ of Algorithm A on G_1, the component on vertex 1 of a random graph $G(n, c/n)$, satisfies*

$$\mathbb{E}(t(G_1)/|G_1) = O(1)\, \mathbb{E} \exp(c(\sqrt{2}-1) \min\{U'W'/n, U'/2\}), \tag{10}$$

with U' and W' given by the random walk RW.

Proof. The running time of Algorithm A on a connected graph with n_1 vertices, m_1 edges and excess $\kappa_1 = m_1 - n_1$is at most

$$O(n_1 2^{\kappa_1/2}). \tag{11}$$

The exponential moments of binomial random variables are simple and well known: If a random variable U has distribution $B(N, p)$, then

$$\mathbb{E}z^U = \sum_{i=0}^{N} \binom{N}{i} z^i p^i (1-p)^{N-i} = (pz + (1-p))^N = (1 + p(z-1))^N \le \exp(p(z-1)N),$$

and in particular,

$$\mathbb{E}\sqrt{2}^U \le \exp((\sqrt{2}-1)Np). \tag{12}$$

Setting $N = \min\{U'W', \binom{U'}{2}\}$ and combining (9), (11), and (12) gives

$$\mathbb{E}(t(G_1)/|G_1|) = O(1)\mathbb{E}(2^{\kappa/2}) \le O(1)\mathbb{E} \exp((\sqrt{2}-1)N\, c/n), \tag{13}$$

Noting that $U' \le n$, and so $\binom{U'}{2}/n \le U'/2$, yields (10). □

In the following, we therefore focus on finding bounds on the probability $\Pr(U, W)$ that the "first" component of a random graph has order U and width W, or, since $(U, W) \preceq (U', W')$ per Claim 15, the corresponding probability $\Pr(U', W')$ for the random walk RW.

We use a version of Chernoff's inequality (see [JŁR00, Theorems 2.1 and 2.8]), which states that for a sum Z of independent 0-1 Bernoulli random variables with parameters $p_1, \ldots, p_n$ and expectation $\mu = \sum_{i=1}^{n} p_i$:

$$\mathbb{P}(Z \geq \mu + t) \leq \exp\left(-t^2/(2\mu + 2t/3)\right) \tag{14}$$

$$\mathbb{P}(Z \leq \mu - t) \leq \exp\left(-t^2/(2\mu)\right). \tag{15}$$

The next lemma describes the probability that U' is large, and has a corollary for the probability that $|G_1|$ is large. (We will not use the corollary, but we state it because it is natural and potentially useful.) Although the proof is framed in terms of the binomial increments $B(n-i, c/n)$ for RW (corresponding to vertex exposures in the random graph), it may also be useful to think in terms of subdividing such an increment into $n-i$ Bernoulli increments $\mathrm{Be}(c/n)$ (corresponding to edge exposures in the random graph). This view will be essential in proving Lemma 19. [1]

This view is illustrated in Figure 2, whose X axis indicates edge exposures j in the augmented graph model, or equivalently the number of Bernoulli random variables exposed in the random walk (a "finer sampling" of the same RW). Since the number of edge exposures between successive deaths shrinks from $n-1$ to $n-2$ etc., the number of deaths (the function $d(j)$) grows super-linearly. (As it happens, $j^{-1}(d)$ is a parabola, i.e., $d(j)$ is a parabola rotated sideways.) The expected number of births grows linearly, $\mathbb{E}X(j) = (c/n)j$, and (for $c = 1$) is tangent to the "death curve" at the origin. The event that the actual number of births equals deaths equals αn means that a corresponding sum of Bernoulli random variables $\mathrm{Be}(c/n)$ equals αn; the individual births comprising this sum describe a random walk, a sample of which is shown in the figure.

Lemma 17. *Given an integer $n > 0$ and a value $c \leq 1 + \lambda n^{-1/3} = 1 + \Lambda$, for any integer $i > 0$, setting $\alpha = i/n$, the time-i widths $W'(i)$ of the random walk RW with parameters c, n satisfy*

$$\Pr(W'(\alpha n) \geq 0) \leq \exp\left(\frac{-3\alpha^3 n(1 - 6\Lambda/\alpha)}{24 - 8\alpha}\right). \tag{16}$$

Proof. By definition, $U' \geq \alpha n$ only if the random walk's width $W'(i)$ at time i is non-negative, so we consider $\Pr(W'(\alpha n) \geq 0)$. Note that $W'(i)$ has distribution

$$\begin{aligned} W'(i) &\sim B\left((n-1) + \cdots + (n-i), \frac{1+\Lambda}{n}\right) - i + 1 \\ &= B\left(ni - \binom{i+1}{2}, \frac{1+\Lambda}{n}\right) - i + 1 \end{aligned}$$

and so $W'(i) \geq 0$ means that

$$B\left(ni - \binom{i+1}{2}, \frac{1+\Lambda}{n}\right) \geq i + 1 = \alpha n + 1. \tag{17}$$

[1] The edge-exposure model was previously used by Spencer in an elegant short paper [Spe97]. Spencer was studying a related problem, calculating $C(n, n+k)$ (the number of connected graphs with k vertices and $n+k$ edges) for k fixed and $n \to \infty$. We will discuss $C(n, n+k)$ in Section 6.

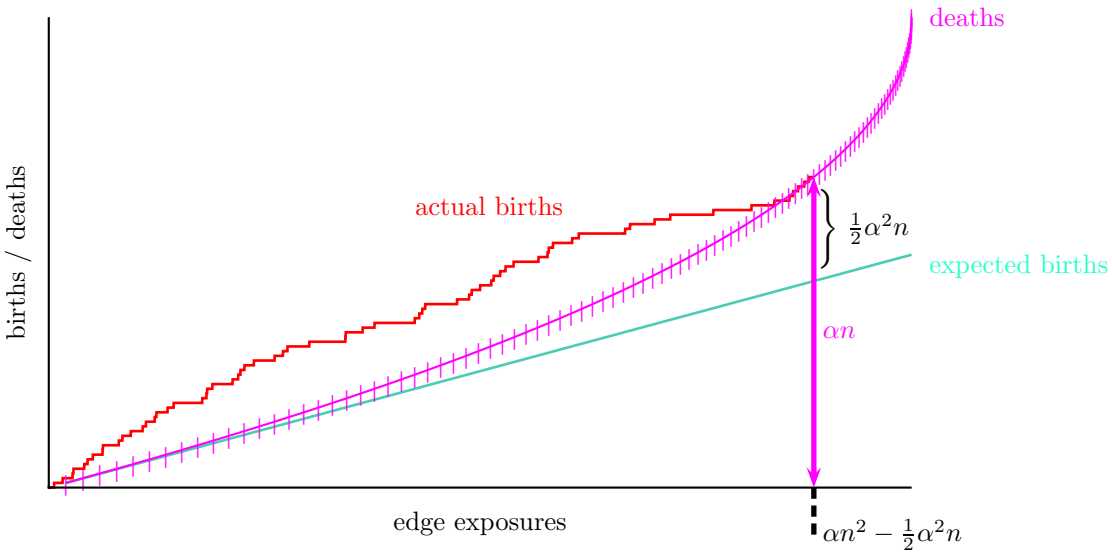


FIGURE 2. Birth / death process described by edge exposures in the augmented graph process, or equivalently by Bernoulli exposures in RW.

This binomial r.v. has expectation

$$\left(\alpha n^2 - \binom{\alpha n + 1}{2}\right) \frac{1+\Lambda}{n} \leq (1+\Lambda)(\alpha - \alpha^2/2)n. \tag{18}$$

For convenience, define $q = \Lambda/\alpha$. Thus if (17) holds, the r.v. exceeds its expectation by at least

$$\alpha^2 n/2 - \Lambda(\alpha - \alpha^2/2)n = \frac{\alpha^2 n}{2}(1 - 2q + \alpha q) \geq 0. \tag{19}$$

Together with (18) and (19), (14) implies that (17) has probability at most

$$\exp\left(\frac{-\alpha^4 n^2 (1-2q+\alpha q)^2/4}{2(1+q\alpha)(\alpha - \alpha^2/2)n + \alpha^2 n(1-2q+\alpha q)/3)}\right). \tag{20}$$

A short calculation shows that this is at most (16). □

Note that this lemma has an immediate consequence for large components in a random graph.

Corollary 18. *Given an integer $n > 0$ and a value $c \leq 1 + \lambda n^{-1/3} = 1 + \Lambda$, for any integer $i > 0$, setting $\alpha = i/n$, the order of the component G_1 containing vertex 1 in a random graph $G(n, c/n)$ satisfies*

$$\Pr(|G_1| \geq \alpha n) \leq \exp\left(\frac{-3\alpha^3 n(1 - 6\Lambda/\alpha)}{24 - 8\alpha}\right).$$

Lemma 19. *Given an integer $n \geq 2$ and a value $c \leq 1 + \lambda n^{-1/3} = 1 + \Lambda$, RW has the property that, for any $\beta \geq \frac{\alpha^2}{8-4\alpha} + \frac{3}{n}$,*

(21)
$$\Pr(W' \geq \beta n \mid W'(\alpha n) = 0) \leq \exp\left(-(3\ln(3) - 2)\ \left(\beta - \frac{\alpha^2}{8-4\alpha} - \frac{3}{n}\right)^2 \frac{n}{\alpha}\right).$$

Proof. Since the width $W'(t)$ decreases by at most 1 per step, if $W'(t) = \beta n$ then it takes additional time at least βn before $W(\cdot) = 0$ is possible, so we may assume $\beta \leq \alpha$.

Switching from the "vertex exposure" to the edge exposure view for the random walk, at the tth step of RW, the number of "edge exposures" is

$$e(t) := (n-1) + \cdots + (n-t) = tn - \binom{t+1}{2},$$

and the total number of births is $Z_1 + \cdots + Z_{e(t)}$, where the Zs are i.i.d. Bernoulli $\mathrm{Be}(c/n)$ random variables.

For the remainder of the proof, equate $i := \alpha n$. Since the number of deaths by the tth vertex exposure is t, and the number of births is $Z_1 + \cdots + Z_{e(t)}$, and we start with one live vertex, the condition $W'(i) = 0$ means that the sequence $Z_1, Z_2, \ldots$ is conditioned by $Z_1 + \cdots + Z_{e(i)} = i - 1$.

For any time $t \leq i$, the number of live vertices is given by

$$W'(t) = Z_1 + \cdots + Z_{e(t)} - (t-1) \tag{22}$$
$$= \left[Z_1 + \cdots + Z_{e(t)} - \frac{e(t)}{e(i)}(i-1)\right] + \left[\frac{e(t)}{e(i)}(i-1) - (t-1)\right]; \tag{23}$$

that is, the gap between the actual number born and its expectation, plus the gap between the expectation and the number of deaths. This view may be more easily apprehended with reference to Figure 3.

The maximum of $W'(t)$, the number of live vertices at any time, is the largest gap between this random walk and the death curve, which can be bounded as the maximum gap between the random walk and a linear interpolation of it plus the maximum gap between the linear interpolation and the death curve. (See Figure 3.) The first of these two gaps is a random quantity governed by Claim 12, while the second is a deterministic function of α and n.

The second term in (23) may be bounded as

$$\begin{aligned}\max_{t \leq i}\left[\frac{e(t)}{e(i)}(i-1) - (t-1)\right] &\leq 1 + \max_{t \leq i}\left[\frac{e(t)}{e(i)}i - t\right] \\ &\leq 1 + i^2/[8n - 4i - 4]\end{aligned}$$

(the maximum occurs at $t = i/2$), and with $i = \alpha n$ and $\alpha \geq 2$, it is easily checked that this is

$$\leq \alpha^2 n/(8 - 4\alpha) + 2.$$

Thus, defining a "discrepancy" δ by $\delta n = \lfloor \beta n - \alpha^2 n/(8-4\alpha) - 2 \rfloor \geq \beta n - \alpha^2 n/(8 - 4\alpha) - 3$, we may rewrite (23) as $W'(t) \leq \left[Z_1 + \cdots + Z_{e(t)} - \frac{e(t)}{e(i)}(i-1)\right] + \beta n - \delta n$

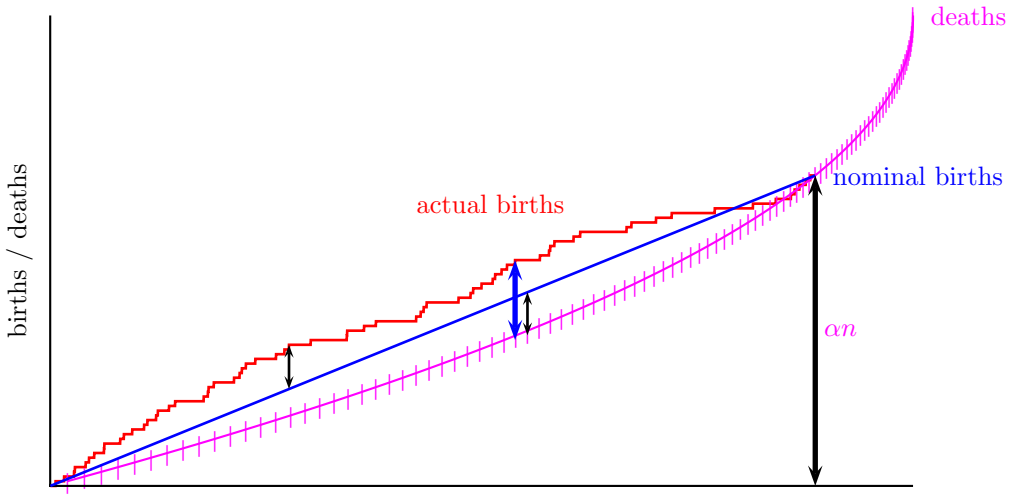


FIGURE 3. Birth / death process

to obtain

$$\Pr(\max_{t\le\alpha n} W'(t) \ge \beta n) \le \Pr\left(\left[\max_{t\le\alpha n} Z_1 + \cdots + Z_{e(t)} - \frac{e(t)}{e(i)}(i-1)\right] \ \ge\ \delta n\right).$$

Recalling that we have conditioned upon $Z_1 + \cdots + Z_{e(i)} = i - 1 = \alpha n - 1$, we apply Theorem 12, with $S = i - 1 = \alpha n - 1$ and $b = \delta n$. (The Claim's "n" is $e(i)$, so its "$p = S/n$" is $\frac{i-1}{in+\binom{i+1}{2}}$, and its hypothesis "$p \le 1/2$" is guaranteed by $n \ge 1$.) Using the notation $B(n,p;k)$ for $B_{n,p}(k)$, this shows the probability to be

$$\le \frac{B(e(i),(i-1)/e(i);\ (i-1)+(2\delta n-2))}{B(e(i),(i-1)/e(i);\ (i-1))}.$$

We already argued that the probability is 0 unless $\beta \le \alpha$, so we may assume $\delta n \le \beta n - 2 \le \alpha n - 2 = i - 2$, and in particular, $2\delta n - 2 < 2(i-1)$: the deviation in question is less than twice the mean. Thus we may apply Remark 13, showing the probability to be

$$\le \exp\left(-(3\ln(3)-2)/4\ (2\delta n-2)^2/(\alpha n-1)\right)$$
$$\le \exp\left(-(3\ln(3)-2)\ \left(\beta - \frac{\alpha^2}{8-4\alpha} - \frac{3}{n}\right)^2 \frac{n}{\alpha}\right).$$

□

6. Remarks on the bounds

That the tail bounds on κ must be done carefully — that the constants count — is illustrated by considering the probability that a large (linear-sized) excess arises simply because the $G(n,1/n)$ random graph has many more edges than expected. Such a graph has $(n+\epsilon n)/2$ edges (rather than the expected $n/2$) with probability

$\exp(-\Theta(\epsilon^2 n))$, and in this case the expected value of κ is $\Theta(\epsilon^3 n)$. Thus for $\epsilon = \Theta(1)$, the probability of excess $\kappa = \epsilon^3 n$ is at least $\exp(-\Theta(\kappa))$, the running time is $\exp(\Theta(\kappa))$, and it becomes critical which of the two constants hidden in the respective Thetas is larger. In particular, it is quite conceivable that it really might be essential to have our running-time bound of $2^{\kappa/2}$ rather than the more naive 2^κ that would be obtained by III-reducing on all vertices of degree 3 or more. Also, good tail bounds on κ will be required even in the fantastically improbable regime $\kappa = \Theta(n)$.[2]

As remarked earlier, tail bounds on κ could also be computed by first-moment methods. Writing $C(u, u+\kappa)$ for the number of connected graphs with u vertices and $u+\kappa$ edges, and applying our algorithm to a random graph $G(n,p)$, the expected time spent on components of order u and excess κ is $2^{\kappa/2}$ times the expected number of them, namely

$$2^{\kappa/2}\binom{n}{u} C(u, u+\kappa) p^{u+\kappa}(1-p)^{u(n-u)+\binom{u}{2}-u-\kappa}. \tag{24}$$

Motivated by the preceding paragraph, we will consider the value of this expression at $p = 1/n$, $u = \epsilon n$ and $\kappa = \epsilon^3 n$; we will fix these values for the remainder of this section.

Wright [Wri80] shows that $C(u, u+\kappa)$ is asymptotically equal to some explicit constant times

$$(A/\kappa)^{\kappa/2} u^{u+(3k-1)/2} \tag{25}$$

where $A = e/12$ and the formula is valid for $1 \ll \kappa \ll u^{1/3}$, i.e., for $\kappa \to \infty$ but $\kappa = o(u^{1/3})$. Since the algorithm's expected time is *equal* to expression (24) summed over all u and κ, and Wright's formula for $C(u, u+\kappa)$ sacrifices only a constant factor, nothing is given up, and this method of calculation must yield a suitable result (a value linear in n) *in the range* where Wright's formula is applicable. Unfortunately, this range does not include the point $u = \epsilon n$ and $\kappa = \epsilon^3 n$.

Similarly, Łuczak [Łuc90] shows that for $0 \ll \kappa \ll u$,

$$1/(e^8\sqrt{\kappa})(A/\kappa)^{\kappa/2} u^{u+(3\kappa-1)/2} \;\le\; C(u, u+\kappa) \;\le\; \sqrt{u^3/\kappa}(A/\kappa)^{\kappa/2} u^{u+(3\kappa-1)/2}.$$

Again, the range of validity of this pair of bounds does not include the point $u = \epsilon n$, $\kappa = \epsilon^3 n$: they would give $A = e/12 + O(\epsilon^2)$, which still does not give the necessary explicit bound for fixed $\epsilon > 0$. It might be possible to extract such a bound, and also to work with all parameter pairs (u, κ) (not just the demonstrative values we chose here), but even then we would be left with the problem of the polynomial leading factors: unless these can be banished, we would be unable to prove that the expected running time is linear rather than merely polynomial.

Bollobás [Bol84] (see also Bollobás [Bol01, V.4]) shows that (25) is a universal upper bound on $C(u, u+\kappa)$ for some universal constant A. Substituting (25) into (24) and evaluating at $p = 1/n$, $u = \epsilon n$ and $\kappa = \epsilon^3 n$ gives (up to small polynomial factors of n and ϵ) $[c(\epsilon)A]^{\epsilon^3 n/2}$, where $c(\epsilon)$ is an easily calculated explicit function. For this method to show that the expected time (for this u and κ) is polynomial in n, we would need a reasonably small upper bound on the constant A.

[2]Were it not for the need to go up into the tails, we could capitalize on results such as those of Aldous [Ald97], that the joint distribution of the largest component's order and excess is asymptotically normal. In Aldous's analysis, Brownian motion plays the same role as our random walk RW.

Bender, Canfield and McKay [BCM90] (in the same journal issue as Łuczak's [Łuc90]) give extremely accurate estimates for $C(u, u+\kappa)$, whose substitution into expression (24) must in principle satisfy our needs. Their formula, though, is rather complex, involves an implicitly defined function, and appears difficult to work with.

That is, while suitable tail bounds could presumably be proved by a first-moment calculation, there are complications. We have therefore chosen, following Karp [Kar90] and others, to adopt a branching-process approach.

The branching-process approach also provides some intuition. It is a classical Erdős-Renyi result [ER61, ER60] that a random graph at the critical density $1/n$ typically has a giant component whose size is $\Theta(n^{2/3})$. Since our component sizes are given as αn, the giant component (which is likely to be the most difficult component for our algorithm to solve, and thus the component we should focus on) would have $\alpha n = \Theta(n^{2/3})$, $\alpha = \Theta(n^{-1/3})$. The inequality (16) "allows" such a component, giving its probability as $\exp(-\Theta(1))$.

In the $\mathcal{G}_{n,p}$ scaling window, with $p = (1+\epsilon)/n = (1+\lambda n^{-1/3})/n$, and $\epsilon = \Theta(n^{-1/3})$, $\lambda = \Theta(1)$, a graph typically has a giant component of order about $\epsilon n = \lambda n^{2/3}$ and with excess $\epsilon^3 n = \lambda^3$, suggesting we consider $\alpha = \lambda n^{-1/3}$ and $\kappa = \lambda^3$. (See Bollobás [Bol01, Chapter VI] and Janson, Łuczak and Rucinski [JŁR00, Chapter 5] for this and related results.) Since our bound suggests that κ might be about $(\alpha n)(\beta n)(c/n)$, this would suggest that the typical "width" (maximum number of live vertices) βn of the giant component would have $\beta = \lambda^2 n^{-2/3}$. Notice that (21) imposes no penalty on β until $\beta = \Theta(\alpha^2)$, and the above values $\alpha = \lambda n^{-1/3}$, $\beta = \lambda^2 n^{-2/3}$ fall just at this point. Furthermore, these values satisfy that $\beta n = \sqrt{\alpha n}$. It is well known that the width of a Poisson branching process which reaches size s is typically $\Theta(\sqrt{s})$ (see for example Devroye [HMRA98]); our branching process is close to Poisson and our width is within a factor of 2 of the width as conventionally defined; so this provides yet another consistency check.

In short, we think that the branching-process analysis offers insight into the likelihood or unlikelihood of observing graph components of given order and excess. Especially if one takes as given the properties of binomial distributions and simple random walks proved in Section 4, the branching process analysis is not unduly complicated, and is entirely self-contained.

7. Assembly

In this section we prove the main theorem.

Theorem 20. *For any $\lambda > 0$ and $c \le (1+\lambda n^{-1/3})/n$, let $G \in \mathcal{G}(n, c/n)$ be a random graph, and let (G, S) be any weighted Max 2-CSP instance over this graph. Then (G, S) can be solved exactly in expected time $O(n)\exp(O(1+\lambda^3))$, and in space $O(m+n)$.*

Proof. Consider Algorithm A applied to the graph G. We calculate the running time as follows: for each component G' of G, let $t(G')$ be the time taken by Algorithm A to find an optimal assignment for G'. For each vertex v of G', we define $t(v) = t(G')/|G'|$. Then the total running time of Algorithm A is $O(m+n) + \sum_{v \in V(G)} t(v)$. Choosing any vertex $v \in V(G)$, we see that the expected running time is at most

$$O((1+c)n) + n\mathbb{E}t(v). \tag{26}$$

It therefore suffices to prove that

$$\mathbb{E}\, t(v) \le \exp(O(1+\lambda^3)) = \exp(O(1+\Lambda^3 n)), \tag{27}$$

where $\Lambda = \lambda n^{-1/3}$.

Recall the random walk RW defined before Lemma 17, with order U' and maximum width W'. The running time of Algorithm A is bounded by (10), and so

$$\mathbb{E}\, t(v) \le \mathbb{E}\, \exp\left[(1+\Lambda)(\sqrt{2}-1)\min\{U'W'/n, n/2\}\right], \tag{28}$$

since each component trivially has order at most n.

We examine the contribution of each possible pair $U' \in \{1,\dots,n-1\}$ and $W' \in \{1,\dots,\frac{1}{2}n^2+O(1)\}$ to the expectation. Specifically, for integers αn and βn, define $E(\alpha,\beta)$ to be the expected time that the algorithm spends on cases with $U' = \alpha n$ and $W' = \beta n$. Thus

$$E(\alpha,\beta) \le \exp\Big((1+\Lambda)(\sqrt{2}-1)\min\{\alpha\beta n, n/2\}\Big)\Pr(U'=\alpha n)\Pr(W'=\beta n|U'=\alpha n).$$

It is enough to prove that

$$\sum_{\alpha n,\beta n} E(\alpha,\beta) \le \exp(O(1+\lambda^3)). \tag{29}$$

We will do this by breaking $[0,1]^2$ into rectangular regions, and summing $E(\alpha,\beta)$ separately over each region.

For any region $R \subseteq [0,1]^2$, $\sum_{(\alpha n,\beta n)\in R} E(\alpha,\beta)$ is at most

$$\exp\Bigg\{\Bigg(\max_R\left[(1+\Lambda)(\sqrt{2}-1)\alpha\beta\right] - \min_R\left[\frac{3\alpha^3(1-6\Lambda/\alpha)}{24-8\alpha}I(\alpha)\right] - \min_R\left[(3\ln(3)-2)\,\left(\beta-\frac{\alpha^2}{8-4\alpha}-\frac{2}{n}\right)^2\frac{1}{\alpha}J(\alpha,\beta-3/n)\right]\Bigg)n\Bigg\}, \tag{30}$$

where $I(\alpha)$ is the indicator function for $(\alpha > 6\Lambda)$ and $J(\alpha,\beta)$ is the indicator function for $\beta \ge \alpha^2/(8-4\alpha)$.

Without loss of generality we restrict Λ to $\Lambda > n^{-1/3}$, since the quantity (30) decreases monotonically for smaller Λ while the target bound $n\exp(O(1+\Lambda^3 n))$ does not decrease below $n\exp(O(1))$. We may also restrict Λ to $\Lambda < 0.01$, since by then the target bound $n\exp(O(0.01^3 n))$ allows us to consider all 2^n possible solutions explicitly.

Given that $\Lambda < 0.01$ and $\alpha \le 1$, on substituting $\beta = \beta' + \frac{3}{n}$ into (30), the first term (taking into account the n inside the exponent) is $\le \exp\{(1+\Lambda)(\sqrt{2}-1)\alpha\beta' n + 3\}$, and we may simply move the $\exp(2)$ outside as an (irrelevant) multiplicative constant. That is, we may simply ignore the $\frac{3}{n}$ in (30), extending the range of summation to $\beta \in [-\frac{3}{n}, 1]$, and so summing

$$\exp\Bigg\{\Bigg(\max_R\left[(1+\Lambda)(\sqrt{2}-1)\alpha\beta\right] - \min_R\left[\frac{3\alpha^3(1-6\Lambda/\alpha)}{24-8\alpha}I(\alpha)\right] - \min_R\left[(3\ln(3)-2)\,\left(\beta-\frac{\alpha^2}{8-4\alpha}\right)^2\frac{1}{\alpha}J(\alpha,\beta)\right]\Bigg)n\Bigg\}. \tag{31}$$

Since the value of (31) for any $\beta > 0$ dominates that for $-\beta$, it suffices to restrict to $(\alpha,\beta) \in [0,1]^2$ and double the result. And, since we are interested in orders of magnitude, the doubling is irrelevant: we are back to summing over $[0,1]^2$.

We now consider four regimes of values (α, β), which together cover the space $[0,1]^2$.

(1) α and β both small: $(\alpha,\beta) \in [0, 1000\Lambda] \times [0, 1000000\Lambda^2]$,
(2) $\beta \le \alpha^2/(8-4\alpha)$ and $\alpha < 1$,
(3) $\beta \ge \alpha^2/(8-4\alpha)$ and $\alpha < 1$,
(4) $\alpha = 1$.

Only the first case contains likely pairs (α,β), and it defines the bound in (29); the remaining cases make a negligible contribution, but are a little trickier to analyze.

Case 1. $R = \big[0,\ 1000\Lambda\big] \times \big[0, 1000000\Lambda^2\big]$. This rectangle R contains likely pairs (α,β), and we simply estimate R's probability as 1. Thus (31) becomes

$$\sum_R E(\alpha,\beta) \le \exp\Big\{(1+\Lambda)(\sqrt{2}-1)\max_R[\alpha]\cdot\max_R[\beta]\cdot n\Big\} = \exp(O(\Lambda^3 n)).$$

Case 2. Instead of considering only $\beta \le \alpha^2/(8-4\alpha)$, we will treat a larger domain, $\beta \le \alpha^2/4$. If $\alpha \le 1000\Lambda$ this falls under Case 1, so we need only consider $\alpha \ge 1000\Lambda$. We cover the space $\alpha \ge 1000\Lambda$, $\beta \le \alpha^2/4$ with rectangles

$$R_i = [{\alpha^\star}_i, 1.01{\alpha^\star}_i] \times [0, ({1.01\alpha^\star}_i)^2/4],$$

with ${\alpha^\star}_i = 1000\Lambda \cdot 1.01^i$, $i = 0,1,\ldots$; it does no harm in our calculations to let i run to infinity. Thus within any such rectangle R, writing $\alpha^\star$ for ${\alpha^\star}_i$, (31) is at most

$$\begin{aligned}
&\exp\left\{\left(\left[1.01^3(1+\Lambda)(\sqrt{2}-1)\cdot\alpha^\star\cdot(\alpha^\star)^2/4\right] - \left[\frac{3{\alpha^\star}^3(1-6\Lambda/\alpha^\star)}{24-8\alpha^\star}\right]\right)n\right\}\\
&\le \exp\left\{\left(\left[1.01^3(1+0.01)(\sqrt{2}-1)/4\right] - \left[\frac{3(1-6/1000)}{24}\right]\right){\alpha^\star}^3 n\right\}\\
&\le \exp\Big(-0.016{\alpha^\star}^3 n\Big).
\end{aligned}$$

Recalling that ${\alpha^\star}_i = 1000\Lambda\cdot 1.01^i$ and that $\Lambda \ge n^{-1/3}$, the contribution to the overall sum (29) is at most

$$\sum_{i=0}^{\infty}\exp\left(-0.016\cdot 1000\Lambda^3\cdot 1.01^{3i}n\right) \le \sum_{i=0}^{\infty}\exp\left(-16\cdot 1.03^i\right) = \Theta(1).$$

Case 3. We divide this case into two sub-cases, $\alpha \le 100\Lambda$ and $\alpha > 100\Lambda$.
Sub-case: $\alpha \le 100\Lambda$**.** If $\beta \le 1000000\Lambda^2$ then Case 1 applies, so we need only consider $\beta > 1000000\Lambda^2$. Break this domain into rectangles

$$R_j = [0, 100\Lambda] \times [{\beta^\star}_j, 1.01{\beta^\star}_j],$$

with ${\beta^\star}_j = 1000000\Lambda^2\cdot 1.01^j$, $j = 0,1,\ldots$; it does no harm to let j run to infinity. For any such rectangle R, writing $\beta^\star$ for ${\beta^\star}_{ij}$, an upper bound on (31) is given by

$$\exp\left\{\left(\left[(1+\Lambda)(\sqrt{2}-1)100\Lambda\cdot 1.01\beta^\star\right] - \left[(3\ln(3)-2)\ \left(\beta^\star - (100\Lambda)^2\right)^2\frac{1}{100\Lambda}\right]\right)n\right\}.$$

Since $\Lambda \le 0.01$ and $(100\Lambda)^2 \le 0.01\beta^\star$, this is

$$\le \exp\left\{\left(\left[(\sqrt{2}-1)1.01^2(100\Lambda)\beta^\star\right] - \left[(3\ln(3)-2)\ 0.99^2\cdot{\beta^\star}^2/100\Lambda\right]\right)n\right\}.$$

Noting that $\beta^\star/100\Lambda \geq 10000\Lambda$, this is

$$\begin{aligned}&\leq \exp\left\{\left(\left[(\sqrt{2}-1)1.01^2 100\right] - \left[(3\ln(3)-2)\ 0.99^2 \cdot 10000\right]\right)\Lambda\beta^\star n\right\}\\ &\leq \exp\left(-10000\Lambda\beta^\star n\right).\end{aligned}$$

Summing over the rectangles R_j with ${\beta^\star}_j = 1000000\Lambda^2 \cdot 1.01^j$, gives a contribution to (29) of at most

$$\sum_{j=0}^{\infty} \exp\left(-10000\Lambda^3 1.01^j n\right) \leq \sum_{j=0}^{\infty} \exp\left(-20000 \cdot 1.01^j\right) = O(1).$$

Sub-case: $\alpha > 100\Lambda$**.** This case, with $\alpha > 100\Lambda$ and $\beta > \alpha^2/(8-4\alpha)$, is the most delicate. Observations of such values α, and of β conditioned upon α, are both unlikely, and we need to keep all three terms in (31).

In this case, we break the domain down into rectangles,

$$R_{ij} = [{\alpha^\star}_i, 1.01{\alpha^\star}_i] \times [{\beta^\star}_{ij}, 1.01{\beta^\star}_{ij}],$$

with ${\alpha^\star}_i = 100\Lambda 1.01^i$, ${\beta^\star}_{ij} = ({\alpha^\star}_i)^2/(8-4{\alpha^\star}_i)\cdot 1.01^j$; we obtain an upper bound by allowing both i and j to run from 0 to infinity. Within any such rectangle R given by $\alpha^\star = {\alpha^\star}_i$ and $\beta^\star = {\beta^\star}_{ij}$, (31) is at most

$$\begin{aligned}&\exp\left\{\left(\left[1.01^2(1.01)(\sqrt{2}-1)\alpha^\star\beta^\star\right] - \left[\frac{3{\alpha^\star}^3(1-6/100)}{24-8\alpha^\star}\right]\right.\right.\\ &\qquad\left.\left. - \left[(3\ln(3)-2)\ \left(\beta^\star - \frac{1.01^2{\alpha^\star}^2}{8-4\cdot 1.01\alpha^\star}\right)^2 \frac{1}{1.01\alpha^\star}\right]\right)n\right\}\\ &=: \exp(-f(\alpha^\star,\beta^\star)n),\end{aligned} \tag{32}$$

and we claim that $f(\alpha,\beta) \geq 0.1\alpha\beta$ for all $0 \leq \alpha \leq 1$, $\alpha^2/(8-4\alpha) \leq \beta \leq 1$.

Verifying this is fairly straightforward. $f(\alpha,\beta)/(\alpha\beta)$ is, for a given α, extremized either by an extreme value of β — $\beta = 1$ or $\beta^\star = (1.01^2\alpha^2)/(8-4\cdot 1.01\alpha)$ — or by the point where its derivative $\frac{d}{d\beta}$ vanishes. The latter is straightforward to compute in closed form: it has one root which is negative (or 0 if $\alpha = 0$), and is therefore infeasible; and another root $\beta = \beta(\alpha)$ which is positive (or 0 if $\alpha = 0$). Substituting $\beta = 1$, $\beta = \alpha^2/(8-4\alpha)$, or $\beta = \beta(a)$ into $f(\alpha,\beta)/(\alpha\beta)$ yields in each case a function which is tractable over $\alpha \in [0,1]$. When the three are graphed, it is easily seen that the smallest value is achieved at $\alpha = 0$, $\beta = \beta(\alpha)$, and here $f > 0.1$. This observation is easily confirmed by elementary calculus.

Given that $f(\alpha,\beta) \geq 0.1\alpha\beta$, summing the contributions of the rectangles R_{ij} to (29) is as in the previous cases. Instead of limiting the summation to $\beta \geq \alpha^2/(8-4\alpha)$, we will actually encompass all $\alpha \geq 100\Lambda$ and all $\beta \geq (100\Lambda)^2/8$, a

much larger domain.

$$\begin{aligned}\sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\exp(-f({\alpha^\star}_i,{\beta^\star}_{ij})n) &\le \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\exp(-0.1{\alpha^\star}_i{\beta^\star}_{ij}n)\\ &\le \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\exp(-0.1\cdot(100\Lambda 1.01^i)\cdot 1.01^j(100\Lambda)^2/8\cdot n)\\ &= \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\exp(-10000\Lambda^3 n 1.01^i 1.01^j)\\ &\le \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\exp(-10000\cdot 1.01^i 1.01^j)\\ &= O(1).\end{aligned}$$

Case 4. If $U' = n$ then $W'(n) \ge 0$, and so we must have had at least $n-1$ births among at most $\binom{n}{2}$ edge probes. By Chernoff, since $c = (1+\Lambda)n \le 1.01n$, and we may assume n is large,

$$\begin{aligned}\Pr(B(\binom{n}{2}, c/n) \ge n-1) &\le \exp\left(-\frac{(0.4949n)^2}{2\cdot 0.505n + 2/3\cdot 0.4949n}\right)\\ &= \exp(-0.1827n).\end{aligned}$$

Conditioning on this event, the number of internal edges is dominated by $B(\frac{1}{2}n^2, 1.01/n)$. Using the $2^{m/4}$ running-time bound, the expected running time is at most

$$\begin{aligned}&\mathbb{E}\exp(-0.1827n)\cdot 2^{(\frac{n}{2}+B(\frac{1}{2}n^2,1.01/n))/4}\\ &= \exp(-0.1827n)\cdot(2^{1/4})^{n/2}\cdot\exp((2^{1/4}-1)\cdot\frac{1}{2}n)\\ &= \exp(n(-0.1827+\frac{1}{2}\cdot\frac{1}{4}\cdot\ln 2+\frac{1.01}{2}(2^{1/4}-1)))\\ &\le \exp(-0.0005n)\\ &= O(1).\end{aligned}$$

It follows that, in each of the five cases above, the contribution of the corresponding $U' = \alpha n$, $W' = \beta n$ sum to expectation $\exp(O(\lambda^3))$, and the result is proved. □

8. Conclusions

In the present paper we focus on Max Cut. Our result for "sparse" instances is strong in that it not only applies right up to $c = 1$, but extends through the scaling window, where $c = 1 + \lambda n^{-1/3}$. We believe that our methods can be extended to Max 2-Sat, but the analysis is certainly more complicated. In fact our results already apply to any Max 2-CSP, and in particular to Max 2-Sat, but only in the regime where there are about $n/2$ clauses on n variables; since it is likely that random Max 2-Sat instances with up to about n clauses can be solved efficiently on average (the Max 2-Sat phase transition occurs around n clauses), our present result for Max 2-Sat is relatively weak. As was the case with Max Cut, it is easy to see that with $m = cn$ and $c < 1/2$ it is almost always easy to solve Max 2-Sat for a random formula with m clauses on n variables: in fact this follows immediately

from the fact that decision 2-Sat is easy, together with the fact that with high probability such a formula is completely satisfiable. The hard part is to show that it is easy not just with high probability but in expectation.

Since Max Cut is in general NP-hard (and even NP-hard to approximate to better than a 16/17 factor [TSSW00]), it would be interesting to resolve whether dense instances of Max Cut as well as sparse ones can be solved in polynomial expected time (thus separating the average-case hardness from the worst-case hardness) or whether random dense instances are hard. Precisely the same questions can be asked about Max 2-Sat, and in both cases we would guess that dense instances are hard, even on average. Such questions are an active research area with no quick resolution in sight.

Acknowledgments

Greg is grateful to Don Coppersmith for helpful and enjoyable conversations.

References

[Ald97] David Aldous, *Brownian excursions, critical random graphs and the multiplicative coalescent*, Ann. Probab. **25** (1997), no. 2, 812–854. MR **98d:**60019

[BBC+01] Béla Bollobás, Christian Borgs, Jennifer T. Chayes, Jeong Han Kim, and David B. Wilson, *The scaling window of the 2-SAT transition*, Random Structures Algorithms **18** (2001), no. 3, 201–256.

[BCM90] Edward A. Bender, E. Rodney Canfield, and Brendan D. McKay, *The asymptotic number of labeled connected graphs with a given number of vertices and edges*, Random Structures Algorithms **1** (1990), no. 2, 127–169. MR **92i:**05108

[Bol84] Béla Bollobás, *The evolution of sparse graphs*, Graph theory and combinatorics (Cambridge, 1983), Academic Press, London, 1984, pp. 35–57. MR **86i:**05119

[Bol01] ———, *Random graphs*, Cambridge Studies in Advanced Mathematics, vol. 73, Cambridge University Press, Cambridge, 2001.

[CGHS] Don Coppersmith, David Gamarnik, Mohammad Hajiaghayi, and Gregory B. Sorkin, *Random* MAX SAT*, random* MAX CUT*, and their phase transitions*, Submitted for publication. 49 pages.

[COMS] Amin Coja-Oghlan, C. Moore, and V. Sanwalani, *Max k-cut and approximating the chromatic number of random graphs*, To appear.

[CR92] Vašek Chvátal and Bruce Reed, *Mick gets some (the odds are on his side)*, 33th Annual Symposium on Foundations of Computer Science (Pittsburgh, PA, 1992), IEEE Comput. Soc. Press, Los Alamitos, CA, 1992, pp. 620–627.

[ER60] P. Erdős and A. Rényi, *On the evolution of random graphs*, Magyar Tud. Akad. Mat. Kutató Int. Közl. **5** (1960), 17–61. MR 23 #A2338

[ER61] ———, *On the evolution of random graphs*, Bull. Inst. Internat. Statist. **38** (1961), 343–347. MR 26 #5564

[FdlV92] Wenceslas Fernandez de la Vega, *On random* 2*-SAT*, Manuscript, 1992.

[GHNR03] Jens Gramm, Edward A. Hirsch, Rolf Niedermeier, and Peter Rossmanith, *New worst-case upper bounds for MAX-2-SAT with an application to MAX-CUT*, Discrete Applied Mathematics (2003), to appear, In Press.

[GKKH77] W. Gellert, H. Kästner, H. Küstner, and M. Hellwich (eds.), *The VNR concise encyclopedia of mathematics*, Van Nostrand Reinhold Co., New York, 1977, With an introduction by Hans Reichardt. MR **83c:**00011

[Goe96] Andreas Goerdt, *A threshold for unsatisfiability*, J. Comput. System Sci. **53** (1996), no. 3, 469–486.

[HMRA98] M. Habib, C. McDiarmid, and J. Ramirez-Alfonsin (eds.), *Probabilistic methods for algorithmic discrete mathematics*, Algorithms and Combinatorics, vol. 16, Springer-Verlag, Berlin, 1998. MR **99i:**05003

[Hoe63] Wassily Hoeffding, *Probability inequalities for sums of bounded random variables*, J. Amer. Statist. Assoc. **58** (1963), 13–30. MR 26 #1908

[JŁR00] Svante Janson, Tomasz Łuczak, and Andrzej Rucinski, *Random graphs*, Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley-Interscience, New York, 2000. MR **2001k:**05180

[Kar90] Richard M. Karp, *The transitive closure of a random digraph*, Random Structures Algorithms **1** (1990), no. 1, 73–93. MR **91j:**05093

[KF02] A. S. Kulikov and S. S. Fedin, *Solution of the maximum cut problem in time* $2^{|E|/4}$, Zap. Nauchn. Sem. S.-Peterburg. Otdel. Mat. Inst. Steklov. (POMI) **293** (2002), no. Teor. Slozhn. Vychisl. 7, 129–138, 183.

[KV02] Michael Krivelevich and Van H. Vu, *Approximating the independence number and the chromatic number in expected polynomial time*, J. Comb. Optim. **6** (2002), no. 2, 143–155.

[Łuc90] Tomasz Łuczak, *On the number of sparse connected graphs*, Random Structures Algorithms **1** (1990), no. 2, 171–173. MR **93a:**05012

[Spe97] Joel Spencer, *Enumerating graphs and Brownian motion*, Comm. Pure Appl. Math. **50** (1997), no. 3, 291–294. MR **97k:**05105

[SS] Alexander D. Scott and Gregory B. Sorkin, *Faster exponential-time algorithms for max 2-sat, max cut, and max k-cut*, In preparation.

[SS03] ———, *Faster algorithms for MAX CUT and MAX CSP, with polynomial expected time for sparse instances*, Proceedings of Random 2003 (Baltimore, MD, 2003), August 2003.

[TCO03] Anusch Taraz and Amin Coja-Oghlan, *Colouring random graphs in expected polynomial time*, Proceedings of STACS 2003, LNCS 2607, 2003, pp. 487–498.

[TSSW00] Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson, *Gadgets, approximation, and linear programming*, SIAM J. Comput. **29** (2000), no. 6, 2074–2097.

[Wri80] E. M. Wright, *The number of connected sparsely edged graphs. III. Asymptotic results*, J. Graph Theory **4** (1980), no. 4, 393–407. MR **82d:**05070

(Alexander D. Scott) Department of Mathematics, University College London, London WC1E 6BT, UK

E-mail address: scott@math.ucl.ac.uk

(Gregory B. Sorkin) Department of Mathematical Sciences, IBM T.J. Watson Research Center, Yorktown Heights NY 10598, USA

E-mail address: sorkin@watson.ibm.com