Do more random bits always help in computation?

Chandan Saha
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur

Abstract

It is a well known fact in computation that the success probability of a randomized algorithm increases with increasing number of independent trials. But this may not be the case when different trials are allowed to interact with and influence each other instead of being totally independent. An example of an algorithm with interacting trials is the "Go With the Winners" algorithm for trees, which uses random bits to find the depth d of a given tree. In this paper we analyze the "Go With the Winners" algorithm and show that (surprisingly) there are instances when raising the number of random bits (trials) from polynomial in depth d to exponential in d decreases the success probability of the algorithm from a constant to an exponentially small probability.

1 Introduction

In the world of computations a random bit is a useful resource for solving problems. There are many problems in computer science for which no efficient deterministic algorithms are known and yet they admit quite efficient randomized solutions. It is natural to ask as to whether increasing the number of random bits always help in computation by reducing the expected running time or the error probability of the algorithm. The answer to this is obviously yes when the rise in the random bits usage takes place through many independent trials of the algorithm. The more the number of trials the less is the probability of failure of the algorithm (in all the trials). However, things get a bit complicated when different trials are allowed to interact and influence each other.

Interactions among different trials of the algorithm can sometimes be very useful to boost the success probability significantly over independent trials. An example of such a case is the "Go With the Winners" algorithm for trees introduced by Aldous and Vazirani [AV94]. "Go With the Winners" (GWW) is a simple 'interactive trials' strategy used to find the deepest node in a tree with large number of nodes which makes the use of a deterministic algorithm infeasible. Aldous and Vazirani showed that the GWW strategy outperforms the 'independent trials' strategy significantly for a large class of trees. However, it was not clear as to whether such 'interactive trials' strategy also has the property that the rising number of trials increases the success probability of the algorithm or that the success probability does not drop significantly with increasing number of trials. In this paper we analyze the GWW algorithm for trees and show that there are instances where the rise in the number of interactive trials from polynomial in depth d of the tree to exponential in d lowers the success probability from a constant to an exponentially small quantity. A reason for showing the analysis here is that it provides some insight to such a drastic fall in performance of the algorithm with increasing usage of random bits which is slightly surprising and counterintuitive. Before we start the main discussion we briefly mention the related works on the GWW strategy in the following paragraph.

The "Go With the Winners" (GWW) strategy for trees was extended by Dimitriou and Impagliazzo [DI96] for the case when the state space is a graph instead of a tree. In a subsequent work [DI98] they applied this strategy to the *graph bisection problem*, where given a graph it is required to partition it into two equal pieces such that the number of edges between the two pieces is the smallest among all possible partitions. The GWW strategy was also used by Peinado [Pei01] for the *clique problem* over random graphs. Other applications can be found in [PL97a], [PL97b], [PL03] and [SKAS05].

2 The "Go With the Winners" Algorithm

Assume that every edge of the input tree is associated with a *transition probability*, such that an algorithm visiting a node of the tree may choose to go to any of its children based on the transition probabilities associated with the edges. A simple algorithm would be to start from the root of the tree and traverse the tree level-wise from a node to a child node till a leaf is reached. Going by the terminology of Aldous and Vazirani [AV94] we call this algorithm Algorithm 0. Several independent executions of Algorithm 0 boosts the success probability of finding a deepest node. However, instead of independent executions, a simple interaction strategy among different executions of this algorithm greatly improves the success probability. A single execution of Algorithm 0 can be abstracted as the traversal of a particle from the root to deeper levels of the tree. The "Go With the Winners" strategy is to start with $B = poly(d)$ particles at root, where d is the depth of the input tree and proceed stage-wise. At every stage all the B particles make independent transitions from a level of the tree to the next level. All particles that are stuck at leaves are then evenly distributed among the particles at non-leaves followed by the start of the next stage. The process repeats till all B particles are at leaves of some level.

Let T be a tree of depth d. Each edge of the tree is associated with a transition probability. For every vertex v in T, $p(v)$ denotes the probability that a particle reaches v when allowed to move freely and independently from the root. If w is a child of v then $p(w|v)$ denotes the transition probability associated with the edge (v, w). For convenience in analysis we will assume that *all transition probabilities of T are greater than or equal to* $\frac{1}{r(d)}$, where $r(.)$ is some fixed polynomial. The probability of reaching the i^{th} level of T in Algorithm 0 is given by $a(i) = \sum_{v \in V_i} p(v)$, V_i is the set of all vertices at the i^{th} level. We call a non-leaf vertex v a *good non-leaf* if there is a path in the tree from v to one of the nodes at the deepest level. Let S^i_{nl} and S^i_g be the set of non-leaves and good non-leaves, respectively at level i. For any subset $S \subset V_i$ of vertices we define $p(S) = \sum_{v \in S} p(v)$.

The following is a modified version of the GWW algorithm presented by Aldous and Vazirani [AV94].

Algorithm *GWW*: To start with, we have $2B$ particles in a repository $\mathcal{R}$. At stage 0, select and remove B particles uniformly randomly from $\mathcal{R}$ and put them at the root of the input tree T. Repeat the following procedure, starting at stage 0 with B particles at the root.

> At stage i we have B^i particles (i to be treated as a superscript), each at some vertex at depth i. If all the particles are at leaves, then stop. Otherwise, some B^i_{nl} particles are at non-leaves and the remaining $B^i_l = B^i - B^i_{nl}$ particles are at leaves. Return the B^i_l particles back to $\mathcal{R}$. To each of the B^i_{nl} particle positions add $\left\lceil \frac{B}{B^i_{nl}} \right\rceil - 1$ more particles; the extra particles being uniformly randomly chosen and removed from the particles in $\mathcal{R}$. Then let each of the $B^i_{nl} \cdot \left\lceil \frac{B}{B^i_{nl}} \right\rceil$ particles move randomly from

its current vertex to a child following the transition probabilities of the edges.

Claim 2.1 *For every $i \in \{0, \ldots, d\}$, Algorithm GWW either stops before stage i, or the number of particles B^i at depth i ranges between B and $2B$.*

Proof: Assume that the algorithm reaches stage i. Then B_{nl}^{i-1} must be greater than zero and $B^i = B_{nl}^{i-1} \cdot \left\lceil \frac{B}{B_{nl}^{i-1}} \right\rceil \geq B$. Inductively, assume that $B \leq B^{i-1} \leq 2B$, implying that $B_{nl}^{i-1} \leq 2B$. If $B_{nl}^{i-1} > B$ then $B^i = B_{nl}^{i-1} \leq 2B$, otherwise if $B_{nl}^{i-1} \leq B$ then $B^i = B_{nl}^{i-1} \cdot \left\lceil \frac{B}{B_{nl}^{i-1}} \right\rceil \leq B_{nl}^{i-1} \cdot \left(\frac{B}{B_{nl}^{i-1}} + 1 \right) \leq 2B$. ■

The following section develops the background for the analysis.

3 A Convenient Perspective

Algorithm *GWW* starts with B particles at level 0. At stage 1 all the B particles are at level 1 with B_{nl}^1 particles at the non-leaves and B_l^1 particles at the leaves. At this point the algorithm makes $k_1 = \left\lceil \frac{B}{B_{nl}^1} \right\rceil$ copies of these particles by adding $\left(\left\lceil \frac{B}{B_{nl}^1} \right\rceil - 1 \right) \cdot B_{nl}^1$ extra particles from the repository $\mathcal{R}$. Equivalently, we may assume that the algorithm spawns k_1 copies of the original tree T at level 1 (as shown in Figure 1) and considers each group of B_{nl}^1 particles independently for transition to level 2. Yet another perspective would be that the trees $T_1 = T, T_2, \cdots, T_{k_1}$

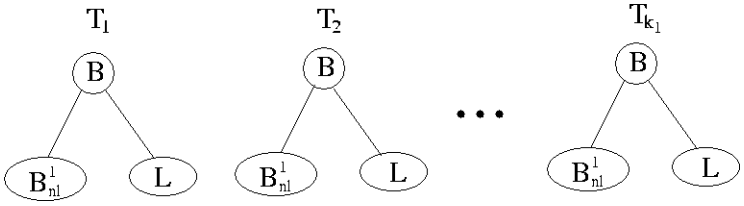

Figure 1: Spawning of trees

are all present from the start of the algorithm (each with B particles at root), and each of them follows the particles of tree T_1 in order to move their own particles from level 0 to level 1. Since k_1 can be at most B, we may assume that all the B trees $T_1, T_2, \ldots, T_B$ are present from the start of the algorithm (each with B particles at root) and all of them follow tree T_1 till level 1, wherefrom they all move their particles independently. At level 1 the algorithm considers only k_1 ($T_1, T_2, \ldots, T_{k_1}$) of these B trees. The dependency among these trees is depicted as a tree $\mathcal{T}$ (see Figure 2),

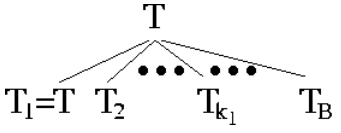

Figure 2: Dependency tree $\mathcal{T}$

At the end of stage 1 all the B trees move their particles independently to level 2 (as shown in Figure 3).

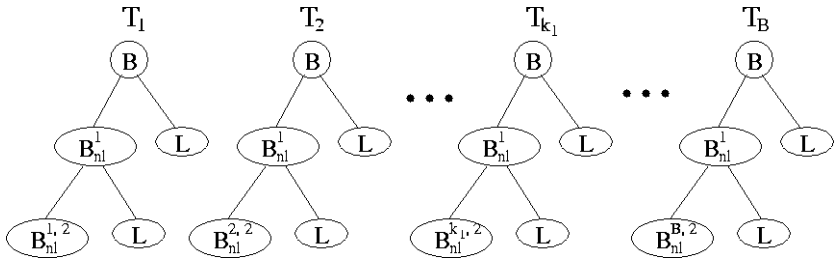

Figure 3: End of stage 1 and start of stage 2

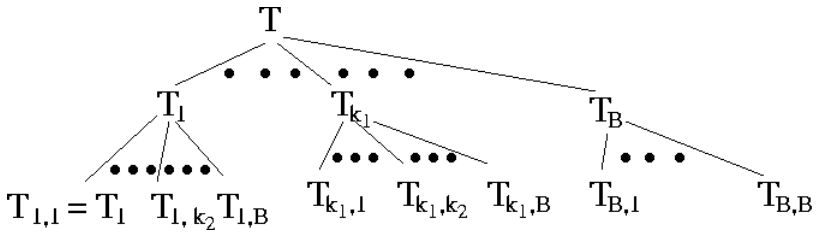

Figure 4: Dependency tree $\mathcal{T}$ till level 2

Therefore, number of particles at the non-leaves of level 2 at the start of stage 2 equals $B_{nl}^2 = \sum_1^{k_1} B_{nl}^{i,2}$. If $B_{nl}^2 > 0$ the algorithm makes $k_2 = \left\lceil \frac{B}{B_{nl}^2} \right\rceil$ copies $(T_{i,1}, T_{i,2}, \ldots, T_{i,k_2})$ of each tree T_i $(1 \leq i \leq k_1)$ and considers them independently for transition to level 3. Since k_2 can be at most B, we may assume that for each i, $1 \leq i \leq B$, all the B trees $(T_{i,1}, T_{i,2}, \ldots, T_{i,B})$ are present from the start of the algorithm each starting with B particles and following the movements of the particles of T_i till level 2. At stage 2 the algorithm considers only k_2 of these B trees, $(T_{i,1}, T_{i,2}, \ldots, T_{i,k_2})$ for each i, $1 \leq i \leq k_1$. As before, the dependency among the trees can be depicted as a tree (see Figure 4).

Extending till stage d, we observe that there are precisely B^d nodes (each node representing a tree) at depth d of the dependency tree $\mathcal{T}$. A tree (or node) at level j of $\mathcal{T}$ follows its parent tree (or node) for particle movements till level j, thereafter it moves its particles independently to the subsequent levels. We can therefore assume that, to start with all the B^d trees are present, each tree follows some other tree based on its dependency given by $\mathcal{T}$ till some level, wherefrom it goes independent. Algorithm GWW considers some subset of these trees at each stage, like k_1 at stage 1, k_1k_2 at stage 2 and so on. Throughout the course of the algorithm, tree T_1 moves its particles independently as if Algorithm 0 is running on T_1. Although a tree T' follows some other tree for particle movements, an observer who only sees T' merely finds Algorithm 0 executing on T'.

4 Analysis of Algorithm GWW

Let $T_1, \ldots, T_{B^d}$ be the B^d trees as discussed in the previous section. Given the $(j-1)$-tuple $\bar{k} = (k_1, k_2, \ldots, k_{j-1})$ one knows exactly which of the trees are considered by algorithm GWW at the end of stage $j-1$. Let $k = \prod_{l=1}^{j-1} k_l$. Without any loss in generality, assume that $T_1, \ldots, T_k$ are the trees considered by the algorithm at the end of stage $j-1$. Denote by X_i^{j-1}, the number of particles at the non-leaves of tree T_i at level $j-1$ and let $X_{g\,i}^{j}$ be the number of particles at the good non-leaves of tree T_i at level j. The number of particles at the good non-leaves of the

given tree at the start of stage j of algorithm GWW is given by X_g^j,

$$X_g^j = X_{g\,1}^j + X_{g\,2}^j + \ldots + X_{g\,k}^j \tag{1}$$

For economy of notation, the symbol $\bar{k}$ inside a probability or the conditional part of an expectation expression will represent the event that $\bar{k}$ is fixed at some specific vector.

Lemma 4.1 $E[X_g^j \mid \mathcal{E}_j, \bar{k}] = k \cdot E[X_{g\,1}^j \mid \mathcal{E}_j, \bar{k}]$.

Proof: From equation (1), by linearity of expectations it follows that,

$$E[X_g^j \mid \mathcal{E}_j, \bar{k}] = \sum_{t=1}^{k} E[X_{g\,t}^j \mid \mathcal{E}_j, \bar{k}]$$

Consider two trees T_{t_1} and T_{t_2} in the dependency tree $\mathcal{T}$, $1 \leq t_1, t_2 \leq k$, where T_{t_2} follows T_{t_1} till level $l \leq j-1$, thereafter they separate out. It is sufficient to observe that for all $x \geq 0$,

$$\Pr\{X_{g\,t_1}^j = x \mid \mathcal{E}_j, \bar{k}\} = \Pr\{X_{g\,t_2}^j = x \mid \mathcal{E}_j, \bar{k}\}$$

■

In fact a similar argument also shows that,

$$E[X_1^{j-1} + X_2^{j-1} + \ldots + X_k^{j-1} \mid \mathcal{E}_j, \bar{k}] = k \cdot E[X_1^{j-1} \mid \mathcal{E}_j, \bar{k}]. \tag{2}$$

Let $\{1, \ldots, B\}$ be the B particles with which GWW starts. We may further assume that these are the particles with which tree T_1 starts executing Algorithm 0. S be the set of particles arriving at the non-leaves of the $(j-1)$-th level of T_1, where $|S| = X_1^{j-1}$.

Assuming that $\mathcal{E}_j$ has occurred, define the variable $\mathcal{Z}(X_1^{j-1}, \bar{k})$ as,

$$\begin{aligned}
\mathcal{Z}(X_1^{j-1}, \bar{k}) &= \Pr\{\text{particle 1 reaches } S_g^j \mid (1 \in S) \wedge (|S| = X_1^{j-1}) \wedge \bar{k}\} \quad \text{if } X_1^{j-1} > 0 \\
&= \frac{p(S_g^j)}{a(j)} \quad \text{else if } X_1^{j-1} = 0
\end{aligned}$$

Lemma 4.2 $E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \,|\, \mathcal{E}_j] = \frac{p(S_g^j)}{a(j)}$ *and* $B \leq E[kX_1^{j-1} \,|\, \mathcal{E}_j] \leq 2B$.

Proof: As before, assume that $\mathcal{E}_j$ has occurred. Then,

$$\begin{aligned}
E[\mathcal{Z}(X_1^{j-1}, \bar{k})] &= \sum_{\bar{k},x} \Pr\{\bar{k} \wedge (X_1^{j-1} = x)\} \cdot \mathcal{Z}(x, \bar{k}) \\
&= \left(\sum_{\bar{k},x>0} \Pr\{\bar{k} \wedge (X_1^{j-1} = x)\} \cdot \mathcal{Z}(x, \bar{k}) \right) + \Pr\{X_1^{j-1} = 0\} \cdot \frac{p(S_g^j)}{a(j)} \\
&= \left(\sum_{\bar{k},x>0} \Pr\{X_1^{j-1} = x\} \cdot \Pr\{\bar{k} \,|\, (X_1^{j-1} = x)\} \cdot \mathcal{Z}(x, \bar{k}) \right) + \Pr\{X_1^{j-1} = 0\} \cdot \frac{p(S_g^j)}{a(j)}
\end{aligned}$$

Note that, for any $x > 0$, $\Pr\{\bar{k} \mid (X_1^{j-1} = x)\} = \Pr\{\bar{k} \mid (X_1^{j-1} = x) \wedge (1 \in S)\}$. This is because,

$$\begin{aligned}
\Pr\{\bar{k} \mid (X_1^{j-1} = x)\} &= \frac{\Pr\{\bar{k} \wedge (X_1^{j-1} = x)\}}{\Pr\{X_1^{j-1} = x\}} \\
&= \frac{\sum_{S_i:|S_i|=x} \Pr\{\bar{k} \wedge (S = S_i)\}}{\sum_{S_i:|S_i|=x} \Pr\{S = S_i\}} \\
&= \frac{\binom{B}{x} \cdot \Pr\{\bar{k} \wedge (S = S_1)\}}{\binom{B}{x} \cdot \Pr\{S = S_1\}} \\
&= \Pr\{\bar{k} \mid (S = S_1)\}
\end{aligned}$$

where S_1 is some fixed set of x elements containing particle 1. The summation in the above expression collapses as $\Pr\{\bar{k} \wedge (S = S_i)\}$ (also $\Pr\{S = S_i\}$) are same for all S_i with size x. Similarly,

$$\begin{aligned}
\Pr\{\bar{k} \mid (X_1^{j-1} = x) \wedge (1 \in S)\} &= \frac{\Pr\{\bar{k} \wedge (X_1^{j-1} = x) \wedge (1 \in S)\}}{\Pr\{(X_1^{j-1} = x) \wedge (1 \in S)\}} \\
&= \frac{\sum_{S_i:(|S_i|=x)\wedge(1\in S_i)} \Pr\{\bar{k} \wedge (S = S_i)\}}{\sum_{S_i:(|S_i|=x)\wedge(1\in S_i)} \Pr\{S = S_i\}} \\
&= \frac{\binom{B}{x-1} \cdot \Pr\{\bar{k} \wedge (S = S_1)\}}{\binom{B}{x-1} \cdot \Pr\{S = S_1\}} \\
&= \Pr\{\bar{k} \mid (S = S_1)\}
\end{aligned}$$

Let z_1 be a boolean variable that is 1 if and only if particle 1 reaches set S_g^j. Then the first part of the expression for $E[\mathcal{Z}(X_1^{j-1}, \bar{k})]$ simplifies as,

$$\begin{aligned}
&\sum_{\bar{k},x>0} \Pr\{X_1^{j-1} = x\} \cdot \Pr\{\bar{k} \mid (X_1^{j-1} = x)\} \cdot \mathcal{Z}(x, \bar{k}) \\
= &\sum_{\bar{k},x>0} \Pr\{X_1^{j-1} = x\} \cdot \Pr\{\bar{k} \mid (X_1^{j-1} = x) \wedge (1 \in S)\} \cdot E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x) \wedge \bar{k}] \\
= &\sum_{x>0} \Pr\{X_1^{j-1} = x\} \cdot \left(\sum_{\bar{k}} \Pr\{\bar{k} \mid (X_1^{j-1} = x) \wedge (1 \in S)\} \cdot E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x) \wedge \bar{k}] \right) \\
= &\sum_{x>0} \Pr\{X_1^{j-1} = x\} \cdot E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x)]
\end{aligned}$$

Therefore, by revealing the fact that the universe is taken to be the event $\mathcal{E}_j$ the expression becomes,

$$\begin{aligned}
&\sum_{x>0} \Pr\{X_1^{j-1} = x \mid \mathcal{E}_j\} \cdot E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x) \wedge \mathcal{E}_j] \\
= &\sum_{x>0} \Pr\{X_1^{j-1} = x \mid \mathcal{E}_j\} \cdot E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x)] \qquad \text{as } (1 \in S) \wedge \mathcal{E}_j = (1 \in S)
\end{aligned}$$

Now note that, for $x > 0$,

$$\begin{aligned}
& E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x)] \\
= &\Pr\{\text{particles 1 reaches } S_g^j \mid (1 \in S) \wedge (X_1^{j-1} = x)\} \\
= &\frac{p(S_g^j)}{a(j)}
\end{aligned}$$

because when taken over the whole universe, it is just Algorithm 0 executing on the B particles of T_1 which makes the transition probability of particle 1 independent of the value of X_1^{j-1}. Therefore,

$$\begin{aligned} E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \mid \mathcal{E}_j] &= \sum_{x>0} \Pr\{X_1^{j-1} = x \mid \mathcal{E}_j\} \cdot \frac{p(S_g^j)}{a(j)} + \Pr\{X_1^{j-1} = 0 \mid \mathcal{E}_j\} \cdot \frac{p(S_g^j)}{a(j)} \\ &= \frac{p(S_g^j)}{a(j)} \end{aligned}$$

The proof of the second statement is simple.

$$\begin{aligned} E[kX_1^{j-1}] &= E[E[kX_1^{j-1}|\bar{k}]] \\ &= E[k \cdot E[X_1^{j-1}|\bar{k}]] \\ &= E[E[X_1^{j-1} + \ldots + X_k^{j-1} \mid \bar{k}]] \text{ , by equation (2)} \end{aligned}$$

The term $X_1^{j-1} + \ldots + X_k^{j-1}$ is the total number of particles at level $j-1$, just before transition to level j. From Claim 2.1, we know that this number is always between B and $2B$. ■

Theorem 4.1 *Assuming that $\mathcal{E}_j$ has occurred,*

$$\begin{aligned} E[X_g^j] &\geq B \cdot \frac{p(S_g^j)}{a(j)} + cov(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1}) \quad \textit{and} \\ E[X_g^j] &\leq 2B \cdot \frac{p(S_g^j)}{a(j)} + cov(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1}) \end{aligned}$$

Proof: Assume that our universe of events is the set of all events where $\mathcal{E}_j$ has occurred and $\bar{k}$ is fixed at some particular vector $(k_1, \ldots, k_{j-1})$. Let $S = \{e_1, e_2, \ldots, e_{|S|}\}$ be the subset of particles from $\{1, \ldots, B\}$ arriving at the non-leaves of level $j-1$ of T_1, where $|S| = X_1^{j-1}$. Then,

$$\begin{aligned} X_{g\,1}^j &= z_{e_1} + z_{e_2} + \ldots + z_{e_{|S|}} \quad \text{where} \\ z_{e_i} &= 1 \quad \text{if } e_i \text{ makes a transition to a good node at level } j \\ &= 0 \quad \text{otherwise} \end{aligned}$$

Therefore,

$$\begin{aligned} E[X_{g\,1}^j|S] &= \sum_{e_k \in S} E[z_{e_k}|S] \\ &= E[z_{e_1}|S] \cdot |S| \quad \text{, since all the particles are identical.} \end{aligned}$$

The above expression makes sense only if we define $E[z_{e_1}|S]$ for $|S| = 0$. But we have full flexibility in doing so, as $E[X_{g\,1}^j|S] = E[z_{e_1}|S] \cdot |S| = 0$ if $|S| = 0$ irrespective of how $E[z_{e_1}|S]$ is defined. So, we make a slight abuse of notation and for any e, $1 \leq e \leq B$ we define $E[z_e|S] = \frac{p(S_g^j)}{a(j)}$ if $|S| = 0$. Therefore,

$$\begin{aligned} E[X_{g\,1}^j] &= E[E[X_{g\,1}^j|S]] \\ &= \sum_{S_i} \Pr\{S = S_i\} \cdot E[z_{e_1(i)}|S_i] \cdot |S_i| \quad \text{,where } e_1(i) \in S_i \text{ if } |S_i| \neq 0, \text{ otherwise } e_1(i) \triangleq 1 \end{aligned}$$

Note that, even in this restricted universe of $\mathcal{E}_j$ and $\bar{k}$, $E[z_{e_1(i)}|S_i]$'s are the same for all sets S_i's with same size. By defining $E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x)]$ as $\frac{p(S_g^j)}{a(j)}$ for $x = 0$ we get,

$$\begin{aligned} E[X^j_{g\,1}] &= \sum_{x \geq 0} \Pr\{X_1^{j-1} = x\} \cdot E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x)] \cdot x \\ &= \sum_{x \geq 0} \Pr\{X_1^{j-1} = x\} \cdot \mathcal{Z}(x, \bar{k}) \cdot x \\ &= E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \cdot X_1^{j-1}] \quad \text{, since the universe fixes } \bar{k}\text{, it is treated as a constant.} \end{aligned}$$

From Lemma 4.1 we get,

$$\begin{aligned} E[X_g^j] &= k \cdot E[X^j_{g\,1}] \\ &= E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \cdot kX_1^{j-1}] \\ \Rightarrow E[X_g^j \mid \bar{k}] &= E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \cdot kX_1^{j-1} \mid \bar{k}] \text{ , revealing the event that } \bar{k} \text{ is fixed} \\ \Rightarrow \quad E[X_g^j\,] &= E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \cdot kX_1^{j-1}] \quad \text{, taking expectations on either side} \\ &= E[\mathcal{Z}(X_1^{j-1}, \bar{k})] \cdot E[kX_1^{j-1}] + cov(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1}) \end{aligned}$$

Therefore, using Lemma 4.2 we get,

$$\begin{aligned} E[X_g^j] &\geq B \cdot \frac{p(S_g^j)}{a(j)} + cov(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1}) \text{ and} \\ E[X_g^j] &\leq 2B \cdot \frac{p(S_g^j)}{a(j)} + cov(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1}) \end{aligned}$$

■

We now show how Theorem 4.1 can be used to find an instance where increasing the number of particles decreases the success probability of GWW drastically.

The term $E[X_g^j|\mathcal{E}_j]$ is precisely decided by the two terms $B \cdot \frac{p(S_g^j)}{a(j)}$ and $cov(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1})$. Given two random variables X and Y, their covariance $cov(X, Y)$ gives a measure of how parameter X tend to differ from its expectation as parameter Y rises above its expectation. The magnitude of the covariance roughly measures the extent to which X differs from its expectation as Y goes above its expectation. As $B \to \infty$, $\mathcal{Z}(X_1^{j-1}, \bar{k}) \to E[\mathcal{Z}(X_1^{j-1}, \bar{k})] = \frac{p(S_g^j)}{a(j)}$ and $kX_1^{j-1} \to E[kX_1^{j-1}]$, which is between B and $2B$, and the term $cov(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1})$ looses its effect on $B \cdot \frac{p(S_g^j)}{a(j)}$. The covariance can only possibly be significant for relatively smaller values of B. Thus we can hope to find the desired example tree by making the term $\frac{p(S_g^j)}{a(j)}$ sufficiently small and also ensuring that the term $cov(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1})$ is largely positive for smaller values of B. To get a better handle on the covariance term we need to understand the effect of kX_1^{j-1} on $\mathcal{Z}(X_1^{j-1}, \bar{k})$.

The parameter $\mathcal{Z}(X_1^{j-1}, \bar{k})$ is fixed at $\frac{p(S_g^j)}{a(j)}$ for $X_1^{j-1} = 0$. Therefore, in order to find the deviation of $\mathcal{Z}(X_1^{j-1}, \bar{k})$ from its expectation $\frac{p(S_g^j)}{a(j)}$ we focus on the case when $X_1^{j-1} > 0$. Since $E[kX_1^{j-1}]$ is always between B and $2B$ a possible way to understand the effect of kX_1^{j-1} on $\mathcal{Z}(X_1^{j-1}, \bar{k})$ is to study the effect of k on the latter parameter. This is because, for $X_1^{j-1} > 0$,

the parameter kX_1^{j-1} generally exceeds $2B$ with rise in k. When $k = 1$, $X_1^{j-1} = B$ and $kX_1^{j-1} = B = poly(d)$, whereas when k is exponentially large and $X_1^{j-1} > 0$ then kX_1^{j-1} is also exponentially large. Therefore, kX_1^{j-1} exceeds its expectation usually for a large enough value of k. The value of k increases as more leaves are encountered by the algorithm. Hence we can hope to make the covariance term positive if there is more chance of reaching a good node in subtrees with large number of leaves. Indeed this understanding leads us to the desired example.

Observation 4.1 *There are trees for which raising the number of starting particles from a polynomial to exponential in depth d lowers the success probability of* GWW *from a constant to an exponentially small number.*

Proof: Consider the example tree in Figure 5. Let $r(\cdot)$ be some fixed polynomial. C is a

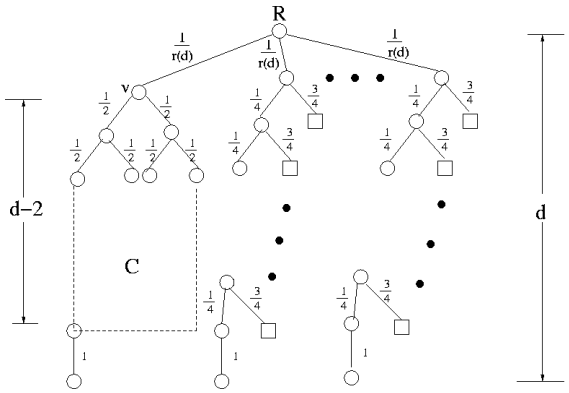

Figure 5: Condition $\mathcal{C}$ is not necessary for GWW to succeed

complete binary tree of depth $d-2$. Consider level $i = d-1$,

$$\frac{p(S_g^i)}{a(i)} < \frac{r(d)}{2(r(d)-1)+2^{d-2}} \leq \frac{r(d)}{2^{d-2}} \tag{3}$$

Suppose that GWW starts with $B = r(d)$ particles. Probability that none of the particles reach vertex v at level 1 is $= \left(1 - \frac{1}{r(d)}\right)^{r(d)} \geq \frac{1}{e} \cdot \left(1 - \frac{1}{r(d)}\right)$. Moreover, if none of the particles reach vertex v then with probability at least $\left(1 - d \cdot \left(\frac{3}{4}\right)^{r(d)}\right)$ GWW succeeds. Therefore, putting the probabilities together we get,

$$Pr\{\text{GWW succeeds with } r(d) \text{ particles}\} \geq \frac{1}{e} \cdot \left(1 - d \cdot \left(\frac{3}{4}\right)^{r(d)}\right) \cdot \left(1 - \frac{1}{r(d)}\right) \geq \frac{1}{4e}.$$

We now show that with $B = d \cdot 2^{\frac{d}{4}} \cdot r(d)$ particles GWW succeeds with probability at most $\frac{d}{2^{\frac{d}{8}-1}}$.

Suppose that the algorithm starts with $B = c \cdot r(d)$, where $c \geq d$ will be fixed later. Applying Chernoff bound we get that, after transition from level 0 to level 1 the probability that number of particles in vertex v is less that $\frac{c}{2}$ is at most $e^{-\frac{d}{8}}$. At the end of stage i, just before transition to level $i+1$, let there be n_i particles at level i within tree C and b particles among other non leaves, where b is between $c \cdot r(d) - n_i$ and $2c \cdot r(d) - n_i$. Assume that $c \cdot r(d) - n_i \geq d$. Again,

from Chernoff bound it follows that the probability that less than $\frac{b}{2}$ particles make transition to leaves at level $i+1$ is at most $e^{-\frac{d}{6}}$. Therefore with probability at least $1-e^{-\frac{d}{6}}$,

$$n_{i+1} \geq \frac{B \cdot n_i}{n_i + \frac{1}{2} \cdot (2c \cdot r(d) - n_i)} \geq \frac{2c \cdot r(d) \cdot n_i}{c \cdot r(d) + n_i} \tag{4}$$

If $n_i = k_i \cdot c \cdot r(d)$ then from (4) we get,

$$k_{i+1} \geq \frac{2k_i}{k_i + 1}$$

Since $k_1 \geq \frac{1}{2r(d)}$ with probability at least $1-e^{-\frac{d}{8}}$, using this as the base case we get,

$$k_i \geq \frac{2^{i-1}}{2^{i-1} + 2r(d) - 1} \quad \text{with probability at least } 1 - i \cdot e^{-\frac{d}{8}}.$$

Therefore, at the i^{th} level number of particles among non-leaves outside C is at most,

$$\frac{(2r(d)-1) \cdot 2c \cdot r(d)}{2^{i-1} + 2r(d) - 1} \quad \text{with probability at least } 1 - i \cdot e^{-\frac{d}{8}}.$$

Choose $c = d \cdot 2^{\frac{d}{4}}$. Within the first $\frac{d}{2}$ levels the number of particles outside C falls below d with probability at least $1 - \frac{1}{2} \cdot d \cdot e^{-\frac{d}{8}}$. Suppose this is the case and B' be the number of particles among the non leaves at level $\frac{d}{2}$. If $B' - d \geq B$ then in the last $\frac{d}{2}$ levels there is no addition of new particles and the d particles outside C reach the last level with probability at most $\frac{d}{2^d}$. Suppose that $B' - d < B$ and let $B' - B = d' < d$. Within the next $\frac{d}{4}$ levels d' particles are lost at leaves with probability at least $1 - d \cdot 2^{-\frac{d}{2}}$. Once the number of particles drops below B, the factor by which each particle is scaled can be at most 2, as $B - d \geq \frac{B}{2}$. Moreover, for the same reason it is also the last time new particles are added. Therefore, with high probability a maximum of $2d$ particles are present outside C at the $\frac{3d}{4}$-th level and no new particles are subsequently added. These particles fail to reach the last level with probability at least $1 - \frac{d}{2^{\frac{d}{2}-1}}$. Therefore, with $B = d \cdot 2^{\frac{d}{4}} \cdot r(d)$ particles GWW succeeds with probability at most $\frac{d}{2^{\frac{d}{8}-1}}$. ■

References

[AV94] David Aldous and Umesh V. Vazirani. "Go With the Winners" Algorithms. *IEEE 35th Annual Symposium on Foundations of Computer Science*, pages 492–501, 1994.

[DI96] Tassos Dimitriou and Russell Impagliazzo. Towards an Analysis of Local Optimization Algorithms. *STOC*, pages 304–313, 1996.

[DI98] Tassos Dimitriou and Russell Impagliazzo. Go with the Winners for Graph Bisection. *SODA*, pages 510–520, 1998.

[Pei01] Marcus Peinado. Go with the Winners Algorithms for Cliques in Random Graphs. In *ISAAC*, pages 525–536, 2001.

[PL97a] Marcus Peinado and Thomas Lengauer. "Go with the winners" Generators with Applications to Molecular Modeling. In *RANDOM*, pages 135–149, 1997.

[PL97b] Marcus Peinado and Thomas Lengauer. Parallel "Go with the Winners" Algorithms in the LogP Model. In *IPPS*, pages 656–664, 1997.

[PL03] Marcus Peinado and Thomas Lengauer. Parallel "go with the winners" algorithms in distributed memory models. *J. Parallel Distrib. Comput.*, 63(9):801–814, 2003.

[SKAS05] László Szirmay-Kalos, György Antal, and Mateu Sbert. Go with the Winners Strategy in Path Tracing. In *WSCG (Journal Papers)*, pages 49–56, 2005.