Go With the Winners: When more Randomness lowers Chance of Success

Chandan Saha*

Department of Computer Science and Engineering
Indian Institute of Technology Kanpur, Kanpur - 208016, India.

Abstract

A study of Aldous and Vazirani's "Go With the Winners" (GWW) algorithm for trees reveals an interesting property - there are trees for which raising the number of random particles from polynomial to exponential in depth d, lowers the success probability of the algorithm from a constant to inverse exponential in d. We analyse the GWW algorithm for the expected case to understand this counterintuitive behavior better. Our analysis sheds some light on the intricate nature of GWW.

Keywords: Analysis of algorithms, Randomized algorithms, Go With the Winners algorithm

---

1. Introduction

The "Go With the Winners" (GWW) algorithm, introduced by Aldous and Vazirani [1], is a simple and effective randomized strategy that successfully finds the deepest node with high probability for a large class of trees where mere 'independent trials' fails to yield good result. The algorithm works by introducing *interactions* among various trials in a very natural way.

Suppose that we are given an input tree whose every edge is associated with a transition probability. The task is to find the depth d of such a tree, which potentially has $\Omega(2^d)$ nodes, using preferably $\mathsf{poly}(d)$ random bits. A simple randomized algorithm starts from the root of the tree and traverses level-wise from node to node based on the transition probabilities. This is called Algorithm 0 in [1]. An execution of this algorithm can be well abstracted by the flow of a particle. A particle visiting a node of the tree chooses to move to one of its children based on the transition probabilities associated with the edges between the node and its children. Instead of using independent executions of the algorithm (or particles in our terminology), the GWW algorithm greatly improves on the success probability by running the different executions simultaneously and making them interact in the following way.

> Start with $B = \mathsf{poly}(d)$ particles at root 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 those particles that are stuck at leaves are 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, which is declared the depth of the tree.

Although very simple to state, analyzing GWW exactly turns out to be quite challenging because of the dependence between positions of different particles. Aldous and Vazirani [1] analyzed an algorithm that supposedly 'approximates' the behavior of GWW and showed that GWW

* ~~Tel: +91-512-259-7123. Fax: +91-512-259-0725~~

Email address: `csaha@cse.iitk.ac.in` (Chandan Saha)

performs well if a certain *imbalance factor* of the tree, that they called κ, is small. To make our discussion more precise, it would help to fix a few notations and conventions.

Notations and conventions - 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). Assume that all transition probabilities of T are greater than or equal to $1/r(d)$, where $r(.)$ is some fixed polynomial. The probability of reaching the i^{th} level of T by a freely moving particle is $a(i) = \sum_{v \in V_i} p(v)$, where V_i is the set of all vertices at the i^{th} level. A non-leaf vertex v is called 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 $p(S) = \sum_{v \in S} p(v)$.

1.1. Objective of our work

Aldous and Vazirani [1] showed that GWW finds a deepest node in a tree with probability $1/4$ using $B = \kappa \cdot \mathsf{poly}(d)$ particles, where κ is a parameter of the tree defined as follows. Define for $0 \leq i < j \leq d$, $\kappa_{i,j} = a(i)/a^2(j) \cdot \sum_{v \in V_i} p(v) a^2(j|v)$. Then κ is defined as, $\kappa = \max_{i,j} \kappa_{i,j}$. The parameter κ is arrived at by analyzing an algorithm that supposedly emulates the behavior of GWW but is comparatively easier to analyze than GWW (see Algorithm 2 in [1]). We show, with an example, that there are natural families of trees with parameter $\kappa = \Omega(2^{d/2})$ where GWW succeeds with high probability using only d particles. In other words, Algorithm 2 of [1] does not fully capture the working of the GWW algorithm.

In an attempt to make the previous sufficient condition (i.e. κ is small) weaker, Roy [2] showed that $\kappa \leq \mathsf{poly}(d)$ implies condition $\mathcal{C} : p(S^i_g)/a(i) \geq 1/\mathsf{poly}(d)$ for all i, $1 \leq i \leq d$. The latter condition roughly corresponds to the fact that there are ‘many’ good non-leaves at every level of the tree. It was conjectured in [2] that $p(S^i_g)/a(i) \geq 1/\mathsf{poly}(d)$ for all i, is a necessary and sufficient condition for GWW to succeed with high probability. We show, with examples, that the above, seemingly natural, condition $\mathcal{C}$ is neither necessary nor sufficient for high success guarantee of GWW. The counterexample to the necessary condition also reveals a surprising property that GWW exhibits. There are trees where GWW succeeds with constant probability using only $\mathsf{poly}(d)$ particles but fails with probability at least $1 - d/2^{d/8}$ when the number of particles is raised to about $2^{d/4}$.

The objective of our work is to better understand these examples by analysing the GWW algorithm. Our analysis finds a relation between the expected number of particles reaching the good nodes of a level (given that the algorithm has reached that level) and the term $B \cdot p(S^i_g)/a(i)$. The latter term is the expected number of particles reaching the good nodes of level i when all the B particles move independently and reach level i. Let $\mathcal{E}_i$ be the event that GWW reaches the i^{th} level and X^i_g be the number of particles among the good nodes of that level. We show that $E[X^i_g|\mathcal{E}_i] \approx B \cdot p(S^i_g)/a(i) + \mathrm{cov}(\mathcal{Z}, \mathcal{Y})$, where $\mathcal{Z}$ and $\mathcal{Y}$ are two random variables such that $\mathcal{Y}$ roughly measures the number of leaves encountered by GWW till the i^{th} level and $\mathcal{Z}$ measures the likelihood of a particle reaching a good node of the i^{th} level for a given $\mathcal{Y}$. In section 6, we discuss how the above expression helps us understand the above-mentioned examples and thereby gain a little more insight into the behavior of the GWW algorithm.

2. The GWW Algorithm

We present a slightly modified version of the GWW algorithm introduced by Aldous and Vazirani [1]. 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 $\lceil B/B^i_{nl} \rceil - 1$ more particles; the extra particles being uniformly randomly chosen and removed from the particles in $\mathcal{R}$. Then let each of the $B^i_{nl} \cdot \lceil B/B^i_{nl} \rceil$ particles move independently randomly from its current vertex to one of its children 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^{i-1}_{nl} must be greater than zero and $B^i = B^{i-1}_{nl} \cdot \lceil B/B^{i-1}_{nl} \rceil \geq B$. Inductively, assume that $B \leq B^{i-1} \leq 2B$, implying that $B^{i-1}_{nl} \leq 2B$. If $B^{i-1}_{nl} > B$ then $B^i = B^{i-1}_{nl} \leq 2B$, otherwise if $B^{i-1}_{nl} \leq B$ then $B^i = B^{i-1}_{nl} \cdot \lceil B/B^{i-1}_{nl} \rceil \leq B^{i-1}_{nl} \cdot \left(B/B^{i-1}_{nl} + 1\right) \leq 2B$. ■

3. The Examples

Example 3.1. *There are trees with parameter $\kappa = \Omega(2^{d/2})$ on which Algorithm* GWW *succeeds with high probability using $B = d$ particles.*

Proof: Consider the tree in Figure 1. Till height $d' = d/2$ it is a complete binary tree. The subtrees rooted at the $2^{d'}$ vertices of level d' are similar except for one, which is a straight path dropping down to level d. Every edge is associated with a transition probability of $1/2$, except for the edges along the straight path, for which the transition probabilities are all one. It is easy to verify that GWW succeeds with probability greater than $1 - d/2^d$ starting with only d particles. Consider the value of $\kappa_{d',d}$. Note that, $a(d') = 1$ and $a(d) = (2^{d'} - 1) \cdot 2/2^d + 1/2^{d'} = (3.2^{d/2} - 2)/2^d$. By the definition of $\kappa_{i,j}$,

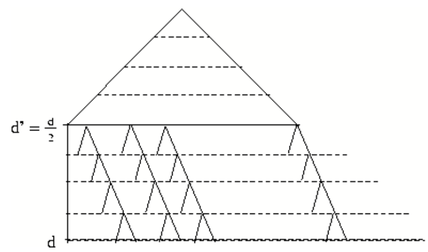

Figure 1: A tree with large κ

$\kappa_{d',d} = 2^{2d}/(3.2^{d/2} - 2)^2 \cdot [2^{-d'} + (1 - 2^{-d'}) \cdot 2^{-2(d-d'-1)}] \geq 1/18 \cdot 2^{d/2}$ and hence $\kappa = \Omega(2^{d/2})$.

Example 3.2. *There are trees in which $p(S^i_g)/a(i) \geq 1/2$ for all $1 \leq i \leq d$, but* GWW *fails with probability at least $1 - c^{-d}$ $(c > 1)$ with* poly(d) *particles.*

Proof: Consider the tree shown in Figure 2. All edge probabilities are $1/2$. C_1, C_2 and C_3 are complete binary trees. If GWW starts with $B =$ poly(d) particles then the probability that a particle reaches the root of C_2 is exponentially small. Therefore, at most $2B$ particles reach the roots of the 'C_3' trees with high probability. However, from there the probability of reaching the last level is at most poly$(d)/2^{d/4-2}$. Hence GWW fails with probability at least $1 -$ poly$(d)/2^{d/4-2}$. It is easy to verify that $p(S^i_g)/a(i) \geq 1/2$ for every i. ■

Example 3.3. *There are trees in which there exist a level i with $p(S_g^i)/a(i) \leq c^{-d}$ $(c > 1)$ and yet* GWW *succeeds with high probability with only* poly(d) *particles.*

Proof: Consider the tree in Figure 3. Let $r(\cdot)$ be some fixed polynomial. C is a complete binary tree of depth $d-2$. For $i = d-1$, $p(S_g^i)/a(i) < r(d)/2^{d-2}$. Suppose that GWW starts with $r(d)$ particles. Probability that none of the particles reach vertex v at level 1 is $(1 - 1/r(d))^{r(d)} \geq e^{-1} \cdot (1 - 1/r(d))$. Moreover, if none of the particles reach vertex v then with probability at least $(1 - d \cdot (3/4)^{r(d)})$ GWW succeeds. It follows that GWW succeeds with probablilty at least $1/4e$. ■

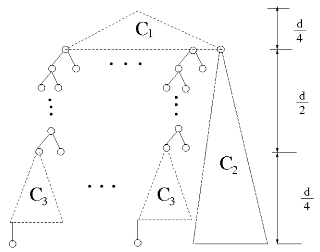

Figure 2: Condition $\mathcal{C}$ is not sufficient

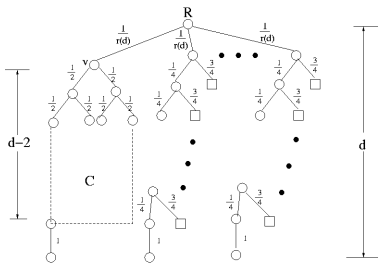

Figure 3: Condition $\mathcal{C}$ is not necessary

The example shown in Figure 3 leads to the following interesting observation.

Observation 3.1. *There are trees for which raising the number of particles from a polynomial to exponential in depth d lowers the success probability of* GWW *from a constant to inverse exponential in d.*

Proof: Consider the tree shown in Figure 3. It can be shown that with $d \cdot 2^{d/4} \cdot r(d)$ particles GWW succeeds with probability at most $d/2^{d/8-1}$. See Appendix A. ■

4. A convenient perspective for Algorithm GWW

Algorithm GWW starts with B particles at level 0. In stage 1, all the B particles move to 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 $k_1 = \lceil B/B_{nl}^1 \rceil$ copies of these particles by adding $\left(\lceil B/B_{nl}^1 \rceil - 1\right) \cdot B_{nl}^1$ 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 4) and considers each group of B_{nl}^1 particles independently for transition to level 2. Yet another perspective is that the trees

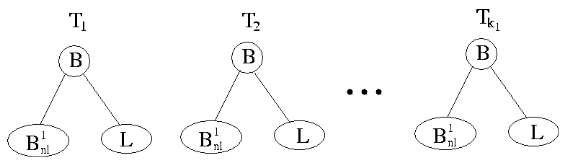

Figure 4: Spawning of trees

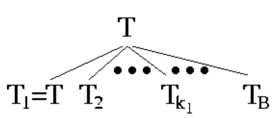

Figure 5: Dependency tree $\mathcal{T}$

$T_1 = T, T_2, \cdots, T_{k_1}$ 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 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 (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 of these B trees, namely $T_1, T_2, \ldots, T_{k_1}$. The dependency among these trees is depicted as a 'metatree' $\mathcal{T}$ in Figure 5. At the end of stage 1 all the B trees move their particles independently to level 2 (as shown in Figure 6).

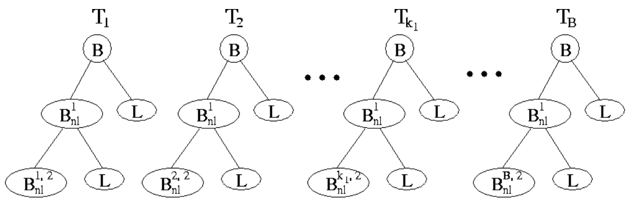

Figure 6: End of stage 1 and start of stage 2

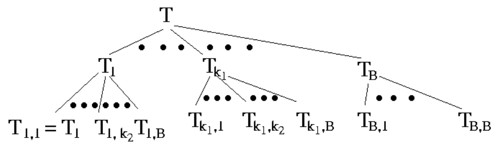

Figure 7: Dependency tree $\mathcal{T}$ till level 2

Therefore, number of particles among 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 $k_2 = \lceil B/B_{nl}^2 \rceil$ 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 particle 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 is depicted in Figure 7. 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, $k_1 k_2$ at stage 2 and so on. Throughout the course of the algorithm, tree T_1 moves its particles independently as if Algorithm 0 of [1] 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 of [1] executing on T'.

5. 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 $k = \prod_{l=1}^{j-1} k_l$ and $T_1, \ldots, T_k$ be 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 input tree at the start of stage j of GWW is given by $X_g^j = X_{g\,1}^j + X_{g\,2}^j + \ldots + X_{g\,k}^j$. 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 to some specific vector.

Claim 5.1. $E[X_g^j \mid \mathcal{E}_j, \bar{k}] = k \cdot E[X_{g\,1}^j \mid \mathcal{E}_j, \bar{k}]$.

Proof: By linearity of expectations, $E[X_g^j \mid \mathcal{E}_j, \bar{k}] = \sum_{t=1}^k E[X_{g\,t}^j \mid \mathcal{E}_j, \bar{k}]$. 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 some level $l \leq j-1$, thereafter they separate out. It is sufficient to observe that for all $x \geq 0$, $\Pr\{X_{g\,t_1}^j = x \mid \mathcal{E}_j, \bar{k}\} = \Pr\{X_{g\,t_2}^j = x \mid \mathcal{E}_j, \bar{k}\}$ ■

Similarly, $E[X_1^{j-1} + X_2^{j-1} + \ldots + X_k^{j-1} \mid \mathcal{E}_j, \bar{k}] = k \cdot E[X_1^{j-1} \mid \mathcal{E}_j, \bar{k}]$. 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 random variable $\mathcal{Z}(X_1^{j-1}, \bar{k})$ as,

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

Lemma 5.1. *$E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \mid \mathcal{E}_j] = p(S_g^j)/a(j)$ and $B \le E[kX_1^{j-1} \mid \mathcal{E}_j] \le 2B$.*

Proof: As before, assume that $\mathcal{E}_j$ has occurred. Then, $E[\mathcal{Z}(X_1^{j-1}, \bar{k})]$ equals

$$\sum_{\bar{k}, x>0} \Pr\{X_1^{j-1} = x\} \cdot \Pr\{\bar{k} \mid (X_1^{j-1} = x)\} \cdot \mathcal{Z}(x, \bar{k}) + \Pr\{X_1^{j-1} = 0\} \cdot p(S_g^j)/a(j).$$

Claim 5.2. *For any $x > 0$, $\Pr\{\bar{k} \mid (X_1^{j-1} = x)\} = \Pr\{\bar{k} \mid (X_1^{j-1} = x) \wedge (1 \in S)\}$.*

Proof: See Appendix A. ■

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

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

Revealing the fact that the universe is the event $\mathcal{E}_j$ the above expression becomes,

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

Now note that, for $x > 0$, $E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x)] = \Pr\{\text{particles 1 reaches } S_g^j \mid (1 \in S) \wedge (X_1^{j-1} = x)\} = p(S_g^j)/a(j)$ 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, $E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \mid \mathcal{E}_j]$ equals $\sum_{x>0} \Pr\{X_1^{j-1} = x \mid \mathcal{E}_j\} \cdot p(S_g^j)/a(j) + \Pr\{X_1^{j-1} = 0 \mid \mathcal{E}_j\} \cdot p(S_g^j)/a(j) = p(S_g^j)/a(j)$.

The proof of the second part is simple since $E[kX_1^{j-1}] = E[E[kX_1^{j-1} | \bar{k}]] = E[k \cdot E[X_1^{j-1} | \bar{k}]] = E[E[X_1^{j-1} + \ldots + X_k^{j-1} \mid \bar{k}]]$. 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. By Claim 2.1, this number is always between B and $2B$. ■

Theorem 5.1. *Assuming that $\mathcal{E}_j$ has occurred,*

$$B \cdot p(S_g^j)/a(j) \leq E[X_g^j] - \mathrm{cov}(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1}) \leq 2B \cdot p(S_g^j)/a(j).$$

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, $X_{g\,1}^j = z_{e_1} + z_{e_2} + \ldots + z_{e_{|S|}}$, where $z_{e_i} = 1$ if e_i makes a transition to a good node of level j and 0 otherwise. Therefore, $E[X_{g\,1}^j|S] = \sum_{e_k \in S} E[z_{e_k}|S] = E[z_{e_1}|S] \cdot |S|$, since all the particles are identical. This 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 $E[z_e|S] = p(S_g^j)/a(j)$ if $|S| = 0$. Therefore, $E[X_{g\,1}^j] = E[E[X_{g\,1}^j|S]] = \sum_{S_i} \Pr\{S = S_i\} \cdot E[z_{e_1(i)}|S_i] \cdot |S_i|$, where $e_1(i) \in S_i$ if $|S_i| \neq 0$, otherwise define $e_1(i) = 1$. 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 of same size. By defining $E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x)]$ as $p(S_g^j)/a(j)$ for $x = 0$ we have, $E[X_{g\,1}^j] = \sum_{x \geq 0} \Pr\{X_1^{j-1} = x\} \cdot E[z_1 \mid (1 \in S) \wedge (X_1^{j-1} = x)] \cdot x = \sum_{x \geq 0} \Pr\{X_1^{j-1} = x\} \cdot \mathcal{Z}(x, \bar{k}) \cdot x = E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \cdot X_1^{j-1}]$ (since the universe fixes $\bar{k}$, it is treated as a constant). From Claim 5.1, $E[X_g^j] = k \cdot E[X_{g\,1}^j] = E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \cdot kX_1^{j-1}]$. Revealing the event that $\bar{k}$ is fixed, $E[X_g^j \mid \bar{k}] = E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \cdot kX_1^{j-1} \mid \bar{k}]$. Taking expectations on either side, $E[X_g^j] = E[\mathcal{Z}(X_1^{j-1}, \bar{k}) \cdot kX_1^{j-1}] = E[\mathcal{Z}(X_1^{j-1}, \bar{k})] \cdot E[kX_1^{j-1}] + \mathrm{cov}(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1})$. The theorem follows from Lemma 5.1. ■

6. Discussion

It is evident from the examples given in Figure 2 and 3 that the term $\mathrm{cov}(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1})$ plays a significant role in deciding $E[X_g^j \mid \mathcal{E}_j]$. However, as $B \to \infty$, $\mathcal{Z}(X_1^{j-1}, \bar{k}) \to E[\mathcal{Z}(X_1^{j-1}, \bar{k})] = p(S_g^j)/a(j)$ and $kX_1^{j-1} \to E[kX_1^{j-1}]$, which is between B and $2B$, and the covariance term looses its effect on $B \cdot p(S_g^j)/a(j)$. The covariance is significant only for relatively smaller values of B, for instance when $B = \mathsf{poly}(d)$. Also, the parameter $\mathcal{Z}(X_1^{j-1}, \bar{k})$ is fixed at $p(S_g^j)/a(j)$ for $X_1^{j-1} = 0$. Therefore, in order to study the deviation of $\mathcal{Z}(X_1^{j-1}, \bar{k})$ from its expectation we should focus on the case when $X_1^{j-1} > 0$. Also, since $E[kX_1^{j-1}]$ is always between B and $2B$ and $X_1^{j-1} \leq B$, to understand the effect of kX_1^{j-1} on $\mathcal{Z}(X_1^{j-1}, \bar{k})$ we should 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 $k = 1$ then $X_1^{j-1} = B$ and $kX_1^{j-1} = B = \mathsf{poly}(d)$, whereas when k is exponentially large and $X_1^{j-1} > 0$ then kX_1^{j-1} is also exponentially large.

Consider the examples given in Figure 1, 2 and 3. In these examples the trees consist of two kinds of substructures; one is a complete binary tree (like C_3 in Figure 2, C in Figure 3 and the straight path in Figure 1), and the other is an elongated structure where each node bifurcates into two children, one of which is a leaf and the other is a non-leaf. For $X_1^{j-1} > 0$, the parameter kX_1^{j-1} rises with rise in k as more and more particles traverse through the second kind of substructures. The value of k rises because more particles are lost as more leaves are encountered along these substructures and therefore more trees in $\mathcal{T}$ are considered by GWW (according to the perspective developed in Section 4).

In Figure 2, the quantity $\mathcal{Z}(X_1^{j-1}, \bar{k})$, which is the probability that a particle reaches a good node at the j^{th} level given it has reached the level, drops to an exponentially small quantity (for $j = d-2$) with the rise in kX_1^{j-1} as more particles travel through the second kind of substructures. This makes the term $\mathrm{cov}(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1})$ largely negative. However, in Figure 1, this drop in probability is only to half and $\text{cov}(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1})$, although negative, is sufficiently small. On the contrary, in Figure 3 as particles travel through the second kind of substructures the probability of a particle reaching a good node increases. Moreover, this change is from an exponentially small quantity (i.e. $p(S_g^j)/a(j)$) to 1/4. This makes $\text{cov}(\mathcal{Z}(X_1^{j-1}, \bar{k}), kX_1^{j-1})$ largely positive for $B = r(d)$. Thus, although $B \cdot p(S_g^j)/a(j)$ is exponentially small, $E[X_g^{j-1}|\mathcal{E}_j]$ is large in this case.

Conclusion - Although the covariance factor helps us understand the example cases better, unlike κ it is not explicitly expressed in terms of the parameters of the input tree. However, on the positive side, our analysis is exactly of the GWW algorithm and not of a 'approximate version' of GWW as is the case in [1]. A future direction of our work would be to investigate if an explicit expression for the covariance factor can be found, and also to bound the variance of the number of particles arriving at the good nodes. Based on the expected case analysis, we are tempted to believe that condition $\mathcal{C}$ is indeed necessary and sufficient for good performance of GWW when the covariance factor is small.

Acknowledgement

I am thankful to Manindra Agrawal, Somenath Biswas and Sudeepa Roy for many insightful discussions during the course of this work.

References

[1] D. Aldous, U. Vazirani, "Go With the Winners" Algorithms, in: FOCS, 1994, pp. 492–501.

[2] S. Roy, On "Go With the Winners" algorithm, Master's thesis, Indian Institute of Technology Kanpur (2006).

Appendix A.

Proof: [Proof of Observation 3.1] Suppose that GWW starts with $B = c \cdot r(d)$. Applying Chernoff bound, after transition from level 0 to level 1 the probability that number of particles in vertex v is less that $c/2$ is at most $e^{-d/8}$. 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, using Chernoff bound it follows that the probability that less than $b/2$ particles go to leaves at level $i+1$ is at most $e^{-d/6}$. Therefore with probability at least $1 - e^{-d/6}$, $n_{i+1} \geq \frac{B \cdot n_i}{n_i + \frac{1}{2} \cdot (2c \cdot r(d) - n_i)} \geq \frac{2c \cdot r(d) \cdot n_i}{c \cdot r(d) + n_i}$. If $n_i = k_i \cdot c \cdot r(d)$ then $k_{i+1} \geq 2k_i/k_i + 1$. Since $k_1 \geq 1/2r(d)$ with probability at least $1 - e^{-d/8}$, using this as the base case we get, $k_i \geq 2^{i-1}/(2^{i-1} + 2r(d) - 1)$ with probability at least $1 - i \cdot e^{-d/8}$. Therefore, at the i^{th} level number of particles among non-leaves outside C is at most, $\frac{(2r(d)-1) \cdot 2c \cdot r(d)}{2^{i-1} + 2r(d) - 1}$ with probability at least $1 - i \cdot e^{-d/8}$. Choose $c = d \cdot 2^{d/4}$. Within the first $d/2$ levels, the number of particles outside C falls below d with probability at least $1 - (1/2) \cdot d \cdot e^{-d/8}$. Suppose this is the case and B' be the number of particles among the non leaves at level $d/2$. If $B' - d \geq B$ then in the last $d/2$ levels there is no addition of new particles and the d particles outside C reach the last level with probability at most $d/2^d$. Suppose that $B' - d < B$ and let $B' - B = d' < d$. Within the next $d/4$ levels, d' particles are lost at leaves with probability at least $1 - d \cdot 2^{-d/2}$. Once the number of particles drops below B, the factor by which each particle is scaled can be at most 2, as $B - d \geq B/2$. 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 $3d/4$-th level and no new particles are subsequently added. These particles fail to reach the last level with probability at least $1 - d/2^{d/2-1}$. Hence, with $B = d \cdot 2^{d/4} \cdot r(d)$ particles $\Pr\{\mathsf{GWW} \text{ succeeds}\} \leq d/2^{d/8-1}$. ■

Proof: [Proof of Claim 5.2] 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,

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

■