J Comb Optim (2016) 32:51–66
DOI 10.1007/s10878-015-9891-9

Performances of pure random walk algorithms on constraint satisfaction problems with growing domains

Wei Xu[1] · **Fuzhou Gong**[2]

Published online: 7 May 2015

Abstract The performances of two types of pure random walk (PRW) algorithms for a model of constraint satisfaction problem with growing domains (called Model RB) are investigated. Threshold phenomenons appear for both algorithms. In particular, when the constraint density r is smaller than a threshold value r_d, PRW algorithms can solve instances of Model RB efficiently, but when r is bigger than the r_d, they fail. Using a physical method, we find out the threshold values for both algorithms. When the number of variables N is large, the threshold values tend to zero, so generally speaking PRW does not work on Model RB. By performing experiments, we show that PRW strategy cannot do better than other fundamental strategies.

Keywords Constraint satisfaction problems · Model RB · Random walk · Local search algorithms

1 Introduction

Constraint satisfaction problems (CSPs) arise in a large spectrum of scientific disciplines, such as computer science, information theory, and statistical physics (Rossi et al. 2006; Lecoutre 2009). A typical CSP instance involves a set of variables and a collection of constraints. Variables take values in a finite domain. Constraints contain

✉ Wei Xu
wxu@buaa.edu.cn

Fuzhou Gong
fzgong@amt.ac.cn

[1] School of Mathematics and Systems Science, Beihang University, Beijing 100191, China

[2] Institute of Applied Mathematics, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China

a few variables and forbid some of their joint values. A solution is an assignment satisfying all the constraints simultaneously. Given a CSP instance, two fundamental scientific questions are to decide the existence of solutions and to find out a solution if it exists. Examples of CSPs are Boolean formula satisfiability (SAT), graph coloring, variants of SAT such as XORSAT, error correction codes, etc.

Random models of CSPs play a significant role in computer science. As instance generators, they provide instances for benchmarking algorithms, help to inform the design of algorithms and heuristics, and provide insight into problem hardness. Classical random CSP models were proposed and denoted by A, B, C and D respectively (Smith and Dyer 1996; Gent et al. 2001), and many alternatives also appeared (Achlioptas et al. 1997; Xu and Li 2000; Smith 2001; Gao and Culberson 2007; Fan and Shen 2011; Fan et al. 2012; Shen and Ren 2014; Zhou et al. 2014).

Model RB is a typical CSP model with growing domains. It was proposed by Xu and Li (2000) to overcome the trivial insolubility of the classical model B, and was proved to have exact satisfiability phase transitions. The instances generated in the phase transition region of Model RB are hard to solve (Xu and Li 2006; Xu et al. 2007) and have been widely used in various kinds of algorithm competitions (see http://www.nlsde.buaa.edu.cn/kexu/ for more information). Model RB develops a new way to study CSPs, especially CSPs with large domains, thus has gotten considerable attention (e.g., Lecoutre 2009; Zhao et al. 2012; Liu et al. 2011; Zhao and Zheng 2011; Richter et al. 2007; Alphonse and Osmani 2008; Jiang et al. 2011; Wang et al. 2011; Liu et al. 2014; Huang and Yin 2014).

Algorithm analysis is a notoriously difficult task. The current rigorous results mostly deal with algorithms that are extremely simple, such as Backtrack-Free algorithms, which assign variables one by one without backtracking (Chao and Franco 1986; Broder et al. 1993). Pure random walk (PRW) algorithm is a process that consists of a succession of random moves. It is relatively simple and has been intensively studied on the k-SAT problem (Alekhnovich and Ben-Sasson 2006; Semerjian and Monasson 2003, 2004; Coja-Oghlan and Frieze 2012; Coja-Oghlan et al. 2009; Schöning 2002, 1999). On k-SAT, A frequently studied PRW algorithm (PRW 2 in the following) is called Walksat. Another reason for PRW algorithm being studied is that random walk is a part of many local search algorithms (Rossi et al. 2006).

In this paper, we study two types of PRW algorithms on Model RB. By experimental methods, threshold phenomenons on performance of these two PRW algorithms are found, just like that of Walksat on k-SAT. Moreover, by a physical method we locate the thresholds for both algorithms, which are $\frac{1-p}{p}\frac{1}{k\ln N}$, with N being the total number of variables, k the number of variables per constraint, p the portion of forbidden joint values per constraint.

This paper is organized as follows. We first give the definition of Model RB and its main properties in Sect. 2. In Sect. 3, we show the threshold behaviors of PRW algorithms by experiments. In Sect. 4, we use a physical method to calculate the thresholds for both algorithms. In Sect. 5, we compare PRW algorithms with other simple algorithms. In Sect. 6, we study the performance of a local search algorithm with random walk. We finally give some concluding remarks in Sect. 7.

2 Model RB

Both classical and revised models of CSPs can be found in (Lecoutre 2009). Here we give the definition of Model RB. Let $k \geq 2$ be an integer. Let $r > 0, \alpha > 0, 0 < p < 1$ be real numbers. Let N be the number of variables and $V = \{\sigma_1, \sigma_2, \ldots, \sigma_N\}$ the set of variables. Each variable takes values from a domain $D = \{1, 2, \ldots, N^\alpha\}$. Each constraint involves k variables and an associated incompatible-set, which is a subset of the Cartesian product D^k. Elements in incompatible-set are called incompatible (forbidden) joint values. Model RB(N, k, r, α, p) is a probability space defined by the folowing steps to generate its instances.

1. We select with repetition $rN \ln N$ constraints independently at random. Each constraint is formed by selecting without repetition k out of N variables independently at random.
2. For each constraint, we form an incompatible-set by selecting without repetition $pN^{\alpha k}$ elements from the Cartesian product D^k independently at random.

A solution is an assignment which satisfies all the constraints. That is to say, the joint values in a solution does not belong to any incompatible-sets of the constraints. The set of all solutions, denoted by $\mathcal{S}$, is a subset of D^N. Let X be the number of solutions, $X = |\mathcal{S}|$. It is easy to see that in Model RB, the expectation of X is

$$\mathbb{E}(X) = N^{\alpha N}(1-p)^{rN \ln N}.$$

Let

$$r_{cr} = -\frac{\alpha}{\ln(1-p)}.$$

If $\alpha > \frac{1}{k}$ and $0 < p < 1$ are two constants, and k and p satisfy the inequality $k \geq \frac{1}{1-p}$, then

$$\lim_{n\to\infty} \Pr(X > 0) = \begin{cases} 1, & r < r_{cr}, \\ 0, & r > r_{cr}. \end{cases}$$

Thus, Model RB has exact satisfiability phase transitions, see (Xu and Li 2000; Zhao and Zheng 2011).

3 Threshold behavior of pure random walk on Model RB

In this section, we study the performances of PRW algorithms on Model RB. By experiments, we find that PRW algorithms exhibit threshold phenomenons, and have different performances before and after the thresholds.

We concentrate on two types of PRW algorithms, called PRW 1 and PRW 2 respectively. In PRW 1, after an initial assignment (to all variables) was given, in each step we randomly reassign a variable from conflict set, where the **Conflict set** is the set of all variables that appear in a constraint that is unsatisfied under the current assignment.

But in PRW 2, we randomly select an unsat-constraint (unsatisfied constraint), then randomly select one of its variables to reassign it.

Algorithm PRW 1

1. Pick up a random assignment. Set up a maximum number of steps.
2. (a) If the conflict set is empty, output the current assignment, terminate the algorithm.
   (b) Otherwise, randomly select a variable in the conflict set, reassign it a value.
3. Repeat step 2, until the repeating time exceeds the maximum step number, then output *fail*.

Algorithm PRW 2

1. Pick up a random assignment. Set up a maximum number of steps.
2. (a) If the current assignment satisfies all constraints, output the current assignment, terminate the algorithm.
   (b) Otherwise, randomly select an unsat-constraint, and randomly select a variable in the constraint, reassign it a value.
3. Repeat step 2, until the repeating time exceeds the maximum step number, output *fail*.

Both algorithms PRW 1 and PRW 2 exhibit threshold phenomenons. The probabilities of getting a solution by algorithm PRW 1 and PRW 2 drop from 1 to 0 dramatically, as shown in Fig. 1. The same threshold phenomenon has been found for Walksat on k-SAT problem (referring to Coja-Oghlan and Frieze 2012), with a conjectured threshold value $\alpha = 2^k/k$.

When $r < r_d$ (r_d is the threshold value) and r is small, the algorithms can find a solution in a short time. Similarly to the description in articles (Semerjian and Monasson 2003, 2004), when r is very small, the hypergraphs (where nodes are variables, and hyperedges are variable sets that form constraints) are mainly composed of isolated trees with very low order. And the average number of solving steps can be obtained by adding up the average number of solving steps on all isolated subgraphs. On a tree with very low order, the average number of solving steps is linear with respect to its order, or to say the number of its nodes. So when r is small, the average number of solving steps is $O(N \ln N)$, as shown in Fig. 2.

When $r > r_d$, variables will be reassigned values again and again, the numbers of unsat-constraints will fluctuate around some plateau value for a long time, as shown in Fig. 3. (Experiments on PRW 1 are similar.) The numbers of unsatisfied clauses exhibits a distribution, as shown in Fig. 4. This is the same as Walksat on k-SAT. Two simple but not rigorous interpretations are as follows. First, the chosen constraint is optimized to be satisfied, but when variables contained in the constraint are reassigned values again for other chosen constraints, the optimization was destroyed. Second, the value of the reassigned variable is optimized, but when variables connected to the this variable are reassigned values, the optimization was destroyed. So when r is big, for

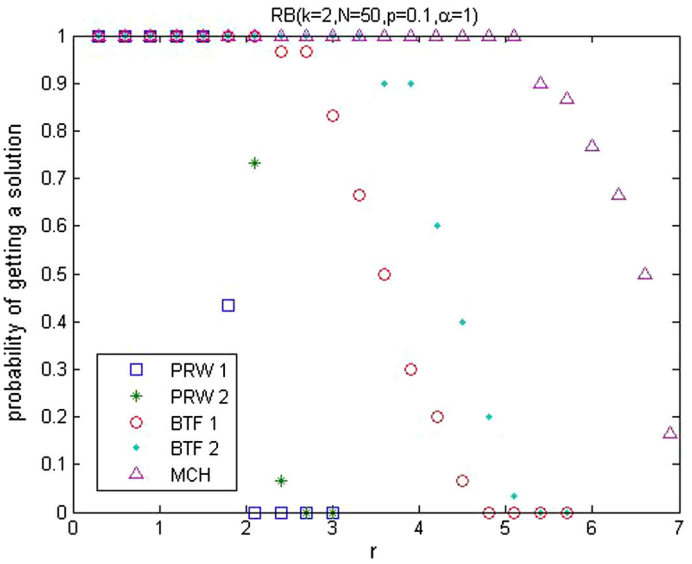

Fig. 1 Probabilities of getting a solution by algorithms PRW 1, PRW 2, BTF 1, BTF 2, and MCH on Model RB. Each point is averaged over 30 runs. The maximum numbers of repeating steps of PRW 1, PRW 2, and MCH are all $40N$

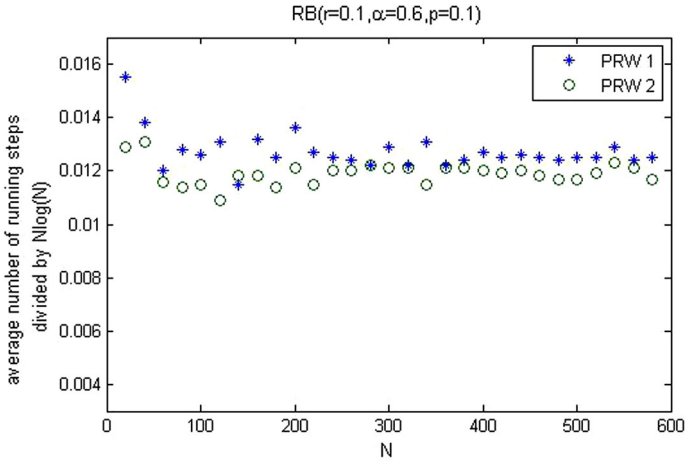

Fig. 2 Average numbers of running steps divided by $Nlog(N)$. Each point is averaged over 100 runs

example $r > r_d$, optimization (or effect of each reassignment) cannot be retained, and algorithms fail.

4 Analysis based on an approximation

The main method used in this section is from theoretical physics, which has been used on k-SAT and other problems by Semerjian et al (Semerjian and Monasson 2003, 2004;

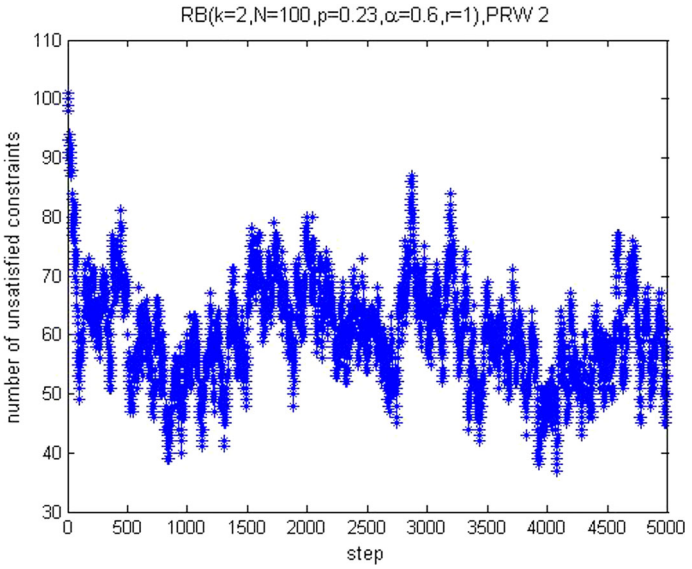

Fig. 3 Numbers of unsat-constraints, PRW 2

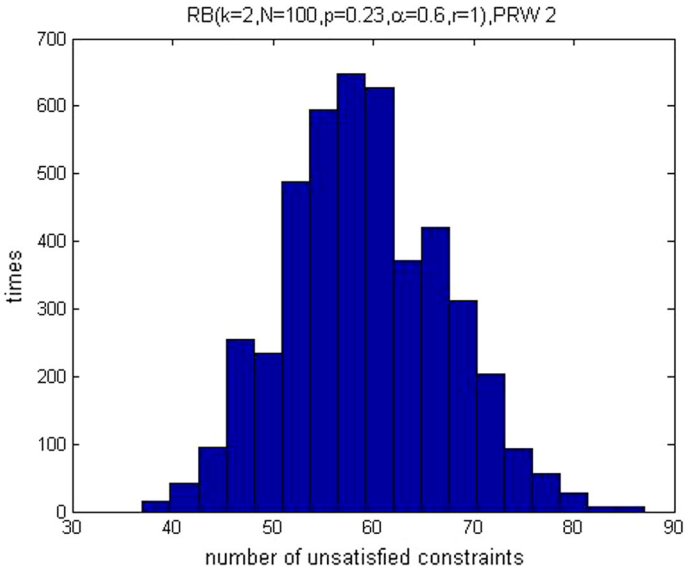

Fig. 4 Histogram of the number of unsat-constraints, where data at the first 500 steps was omitted

Barthel et al. 2003). It is not a rigorous method, since an approximation is utilized, but remarkable results have been gotten on k-SAT problem and XORSAT problem with this method. For more background of this method, we refer to Semerjian and Monasson (2003).

Approximation At each step, before a reassignment, we treat the situation at that time as a typical situation, featured by the number of its unsat-constraints. A typical situation featured by M_0 means that, $M = rN \ln N$ constraints are randomly selected (as the step 1 of Model RB definition), then M_0 unsat-constraints are randomly chosen from the M constraints. When we reassign a variable a random value from its domain, all constraints connected to it become unsatisfied with probability p, satisfied with probability $1 - p$. Then the solving process becomes a Markov chain using numbers of unsat-constraints as its state space, and the transition probability from M_0 to M_0' is the probability that the typical situation featured by M_0 have M_0' unsat-constraints after a step (a reassignment).

First, we study the transition probability from M_0 to M_0'. We will choose a variable to reassign from typical situation featured by M_0. Let $p(Z_1)$ be the probability that a variable with Z_1 unsat-constraints will be chosen, $p(Z_1)$ depends on algorithms and will be discussed below.

The probability that Z_2 constraints become satisfied from being unsatisfied after reassignment is

$$\sum_{Z_1=0}^{M_0} p(Z_1)\binom{Z_1}{Z_2}(1-p)^{Z_2}(p)^{Z_1-Z_2} \triangleq p(Z_2).$$

This is because that Z_2 obeys binomial distribution when Z_1 is fixed.

When $M_0 > 0$, the probability that Z_3 constraints become unsatisfied from being satisfied is

$$\binom{M-M_0}{Z_3}\left(\frac{pk}{N}\right)^{Z_3}\left(1-\frac{pk}{N}\right)^{M-M_0-Z_3} \triangleq p(Z_3). \tag{1}$$

This is because that each of the $M - M_0$ feasible satisfied constraints connects to the reassignment variable with probability $\frac{k}{N}$, and each of the connecting constraints becomes unsatisfied from being satisfied with probability p.

When $M_0 = 0$, we have that $Z_3 = 0$ with probability 1, this is because when $M_0 = 0$, algorithms end.

For a fixed M_0, random variable $M_0' - M_0$ is the sum of two independent random variables Z_3 and $-Z_2$, $M_0' - M_0 = Z_3 - Z_2$,

$$\mathbb{E}(M_0' - M_0) = \mathbb{E}(Z_3) - \mathbb{E}(Z_2). \tag{2}$$

As Z_3 ranges from 0 to M_0, Z_2 ranges from 0 to $M - M_0$, we have that $M_0' - M_0$ ranges from $-M_0$ to $M - M_0$, M_0' ranges from 0 to M, and

$$\sum_{M_0'=0}^{M} p(M_0' - M_0) = 1. \tag{3}$$

Specifically, the transition probability from M_0 to M_0', denoted by $A_{M_0,M_0'}$, equals to the probability $p(M_0' - M_0)$,

$$A_{M_0,M_0'} \equiv p(M_0' - M_0) = \sum_{Z_2=0}^{M_0} \sum_{Z_3=0}^{M-M_0} p(Z_2)p(Z_3)1_{M_0'-M_0+Z_2-Z_3},$$

where

$$1_X = \begin{cases} 1, \text{ if } X = 0, \\ 0, \text{ otherwise.} \end{cases}$$

The initial distribution, i.e. the probability that at time 0 the typical situation is featured by M_0, is

$$\Pr[M_0, 0] = \binom{M}{M_0} p^{M_0}(1-p)^{M-M_0}. \tag{4}$$

Iteratively, the probability that at time $T+1$ the typical situation is featured by M_0' is

$$\Pr[M_0', T+1] = \sum_{M_0=0}^{M} A_{M_0 M_0'} \Pr[M_0, T].$$

Criterion If $\Pr[0, T]$ is 0 (almost) in a long time, it is in the phase after the threshold; if $\Pr[0, T]$ becomes positive from 0 in polynomial time, it is in the phase before the threshold. Denote the average fraction of unsat-constraints at time $T = tM$ by $\varphi(t)$,

$$\varphi(t) = \frac{1}{M} \sum_{M_0=0}^{M} M_0 \Pr[M_0, T = tM].$$

In the beginning, $\Pr[0, T] = 0$ (almost), but if $\varphi(t)$ becomes 0 in polynomial time, then $\Pr[0, T]$ will become positive from 0; if $\varphi(t)$ is always positive, then the number of unsat-constraints will fluctuate around some plateau value, and $\Pr[0, T]$ will always be 0 (almost). The criterion is whether $\varphi(t)$ becomes 0 in polynomial time, when we suppose

$$\Pr[0, T] = 0, \sum_{M_0=1}^{M} \Pr[M_0, T] = 1. \tag{5}$$

4.1 Analysis on PRW 1

When N is large, we might as well say

$$\frac{d\varphi}{dt} = \left(\varphi\left(t + \frac{1}{M}\right) - \varphi(t)\right)/(1/M),$$

then

$$\begin{aligned}\frac{d\varphi}{dt} &= \sum_{M_0'=0}^{M} M_0' \Pr[M_0', tM+1] - \sum_{M_0=0}^{M} M_0 \Pr[M_0, T = tM] \\ &= \sum_{M_0'=0}^{M} M_0' \left(\sum_{M_0=0}^{M} p(M_0' - M_0) \Pr[M_0, T] \right) - \sum_{M_0=0}^{M} M_0 \Pr[M_0, T] \\ &= \sum_{M_0=0}^{M} \Pr[M_0, T] \left(\sum_{M_0'=0}^{M} p(M_0' - M_0)(M_0' - M_0) \right) \\ &= \sum_{M_0=0}^{M} \Pr[M_0, T] \left(\mathbb{E}(M_0' - M_0) \right) \\ &= \sum_{M_0=0}^{M} \Pr[M_0, T] \left(\mathbb{E}(Z_3) - \mathbb{E}(Z_2) \right), \end{aligned} \tag{6}$$

where the third equality is because of (3), the last equality is because of (2).

$\mathbb{E}(Z_3)$ is the average number of constraints becoming unsatisfied from being satisfied, referring to formula (1) when $M_0 > 0$,

$$\mathbb{E}(Z_3) = \sum_{Z_3=0}^{M-M_0} p(Z_3) Z_3 = \frac{k}{N}(M - M_0)p. \tag{7}$$

$\mathbb{E}(Z_2)$ is the average number of constraints becoming satisfied from being unsatisfied,

$$\mathbb{E}(Z_2) = \sum_{Z_2=0}^{M_0} p(Z_2) Z_2 = \mathbb{E}(Z_1)(1-p). \tag{8}$$

For PRW 1, we randomly select a variable to reassign from the conflict set. So when $M_0 > 0$,

$$\mathbb{E}(Z_1) = \mathbb{E}\left(\frac{M_0 k}{NX}\right),$$

where X is the fraction of not empty variables (connecting to at least an unsat-constraint), when we throw M_0 unsat-constraints to N variables. This is because that in the sub-situation "X", NX is the cardinal number of the conflict set, and $M_0 k$ is the total number of connections between variables and unsat-constraints. We have $\mathbb{E}(Z_1) \geq 1$.

$$\mathbb{E}(X) = 1 - (1 - k/N)^{M_0}.$$

When $M_0 = \beta N$, $\beta \to 0$, we have $\mathbb{E}(X)/\beta \to k$. Considering of tail bounds theory for occupancy problem (Kamath et al. 1995), it's easy to prove

$$\mathbb{E}(Z_1) \to \frac{M_0 k}{N\beta k} = 1, as\ \beta \to 0. \tag{9}$$

From (5) to (8), we have

$$\begin{aligned}\frac{d\varphi}{dt} &= \sum_{M_0=1}^{M} \Pr[M_0, T]\frac{k}{N}(M - M_0)p - \sum_{M_0=1}^{M} \Pr[M_0, T]\mathbb{E}(Z_2) \\ &= \frac{krN\ln N}{N}(1-\varphi)p - \sum_{M_0=1}^{M} \Pr[M_0, T]\mathbb{E}(Z_1)(1-p) \\ &= -(1-p) + kr\ln N(1-\varphi)p - \sum_{M_0=1}^{M} \Pr[M_0, T](\mathbb{E}(Z_1) - 1)(1-p). \end{aligned} \tag{10}$$

Let

$$r_d = \frac{1-p}{p}\frac{1}{k\ln N}.$$

For $r < c \cdot r_d$, where $c < 1$ is a constant, from (10) we have

$$\frac{d\varphi}{dt} < -(1-c)(1-p).$$

From (4), we have $\varphi(t = 0) = p$. So when $r < c \cdot r_d$, φ becomes 0 before $t = \frac{p}{(1-c)(1-p)}$. For $r > c \cdot r_d$, where $c > 1$ is a constant, we extra suppose that the last term of (10) is dominated by M_0 around its average. When $\beta = M_0/N = \varphi M/N = \varphi r \ln N$ is near 0, by (9) we can see the last term of (10) is near 0, then $\frac{d\varphi}{dt} > 0$. Then φ is always positive.

According to the criterion, the threshold value of PRW 1 is $r_d = \frac{1-p}{p}\frac{1}{k\ln N}$.

4.2 Analysis on PRW 2

For PRW 2, when $M_0 > 0$, the probability that a variable with Z_1 unsat-constraints are chosen is

$$p(Z_1) = \binom{M_0 - 1}{Z_1 - 1}\left(\frac{k}{N}\right)^{Z_1 - 1}\left(1 - \frac{k}{N}\right)^{M_0 - Z_1}.$$

This is because that in algorithm we randomly select an unsat-constraint, and a variable in the constraint, so each of other $M_0 - 1$ unsat-constraints connects to the variable with probability $\frac{k}{N}$. Therefore when $M_0 > 0$,

$$\mathbb{E}(Z_1) = 1 + (M_0 - 1)k/N. \tag{11}$$

Similarly, for PRW 2, from (5) to (11) we know

$$\begin{aligned}\frac{d\varphi}{dt} &= \sum_{M_0=1}^{M} \Pr[M_0, T]\left(\frac{k}{N}(M - M_0)p - (1-p)(1 + (M_0 - 1)\frac{k}{N})\right) \\ &= -(1-p) + krp\ln N + \frac{k}{N}(1-p) - k\varphi rlnN.\end{aligned}$$

Solving this first-order linear differential equation with the initial condition $\varphi(t = 0) = p$, we get

$$\varphi(t) = p + \frac{1 - p - k(1-p)/N}{rk\ln N}(e^{-rkt\ln N} - 1).$$

Solving equation $\lim_{t\to\infty}\varphi(t) = 0$ of variable r, we have

$$p - \frac{1 - p - k(1-p)/N}{rk\ln N} = 0,$$

we have

$$r = \left(1 - \frac{k}{N}\right)\frac{1-p}{p}\frac{1}{k\ln N} \triangleq r'_d.$$

For $r < cr'_d$, where $c < 1$ is a constant, $\lim_{t\to\infty}\varphi(t) < 0$. Function $\varphi(t)$ decreases and becomes 0 before $t = \frac{p}{(1-c)(1-p)(1-k/N)}$. For $r > cr'_d$, where $c > 1$ is a constant, $\lim_{t\to\infty}\varphi(t) > 0$.

Thus, the threshold value on PRW 2 is

$$r'_d \approx r_d = \frac{1-p}{p}\frac{1}{k\ln N}.$$

4.3 A note

The estimates of threshold values from numerical simulations are always larger than the calculated one. Taking RB($k = 2, N = 350, p = 0.2, \alpha = 0.5$) as an example, the calculated value is $r_d = r'_d = 0.34$, but the simulated value is 0.43 for PRW 1, and 0.51 for PRW 2. However, the simulated values always fall into the region $(r_d, 2r_d)$, so the theoretically calculated values r_d and r'_d reveal the positions of the real threshold values successfully.

5 Compare with other simple algorithms

In this part we compare PRW 1, PRW 2 with other simple algorithms, including Backtrack-Free 1 (BTF 1), Backtrack-Free 2 (BTF 2), and Min-Conflicts Heuristic (MCH).

Algorithm BTF 1

1. Assign σ_1 a random value.
2. For each σ_2, ..., σ_N successively do step 3.
3. (a) Assign the current variable a random value.
   (b) If the current variable violates some constraints with already assigned variables, record the value, reassign the current variable an unrecorded value; else skip (c).
   (c) Repeat (b) until all values in domain are recorded, output *fail*, terminate the algorithm.
4. Output the assignment we obtained.

Algorithm BTF 1 assigns variables from σ_1 to σ_N one by one without backtracking. This algorithm performs badly on k-SAT. But when parameter r satisfies

$$r + \frac{2/3 + \sqrt{4/9 + 8rk}}{2k} < \frac{\frac{1}{k}\alpha}{-\ln(1-p)}, \tag{12}$$

the successful probability of BTF 1 on Model RA (almost equivalent to Model RB) goes to 1 as N grows (Xu 2014). When N is large, the range of r satisfying (12) is much wider than the range $(0, r_d)$, so BTF 1 is much better than PRW on Model RA and RB.

Algorithm BTF 2

1. Let $j = N$. For $i = 0 : 1 : N$, do step 2.
2. (a) If there is a variable which belongs to no more than i constraints, let $\sigma_{\pi(j)}$ be this variable, $j = j - 1$. If $j = 0$, go to step 3.
   (b) Repeat (a) until no such variable exists.
3. According to the new order $\sigma_{\pi(1)}$, ..., $\sigma_{\pi(N)}$, run Algorithm BTF 1.

Algorithm BTF 2 does a permutation step on variables, then according to the new order assigns variables one by one without backtracking. The new order increases the successful probability, and BTF 2 is better than BTF 1. Taking advantage of the discussion in literature (Smith 2001), it can be shown that when parameter r satisfies

$$r < \frac{\frac{1}{k}\alpha}{-\ln(1-p)},$$

the successful probability of BTF 2 on Model RA goes to 1, as N grows.

Algorithm MCH

1. Pick up a random assignment. Set up a maximum number of steps.
2. (a) If the conflict set is empty, output the current assignment, terminate the algorithm; else randomly select a variable from the conflict set.
   (b) Reassign the selected variable a value from its domain which minimizes the number of unsat-constraints. If there are more than one such values, randomly select one from them.
3. Repeat step 2, until the maximum number of steps, then output *fail*.

Algorithm MCH is a simple local search algorithm, it begins with an initial assignment, then does local search steps. In each local search step the selected variable is assigned to a value which minimizes the number of unsat-constraints, and this algorithm has backtracks, so it should perform better than BTF 1, BTF 2.

We compare these algorithms with PRW on Model RB, as shown in Fig. 1. BTF and MCH are suitable for more values of parameter r. This result supports the proposed conclusion that PRW is not a suitable strategy for Model RB.

6 Different random walk steps in WMCH on Model RB

Algorithm Min-Conflicts Heuristic with Random Walk (WMCH) is a local search algorithm, which combines Random Walk and MCH. In each search step, for probability probability of p $(0 < p < 1)$, WMCH runs a random walk step, and for probability probability of $1 - p$, WMCH runs a min-conflicts heuristic step.

Step 2 of PRW 1, step 2 of PRW 2, and step Blind Local Search will be adopted as the random walk step in WMCH respectively.

Step blind local search Randomly select a variable from the set $\{\sigma_1, ..., \sigma_N\}$, and randomly reassign it a value from its domain. If the current assignment satisfies all constraints, output the current assignment, terminate the algorithm.

Blind Local Search searches the feasible solution space blindly, which is a bad strategy. The expected number of steps to find a solution (after an initial assignment is randomly given) is $(1 - p)^{-rN \ln N}$, which is a very big number even when $r = r_d = \frac{1-p}{p} \frac{1}{k \ln N}$.

Algorithm WMCH 1,2,3

1. Pick up a random assignment. Set up a maximum number of steps, and a number p $(0 < p < 1)$.
2. (a) With probability p, do a random walk step (For WMCH 1, it's step 2 of algorithm PRW 1. For WMCH 2, it's step 2 of algorithm PRW 2. For WMCH 3, it's step Blind Local Search.).
   (b) With probability $1 - p$, do step 2 of algorithm MCH.
3. Repeat step 2, until the maximum number of steps, then output *fail*.

Experimental result indicates that WMCH 1,2 have the same effect as WMCH 3, and without doubt they are better than MCH, as shown in Fig. 5. Being random walk

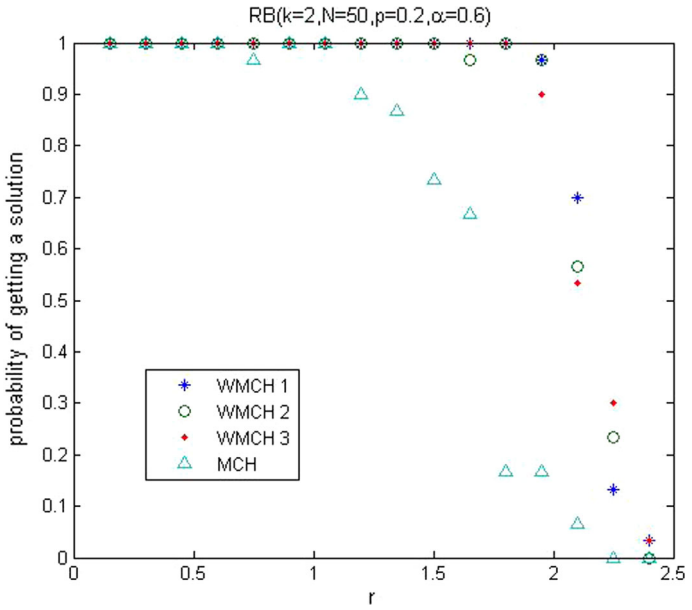

Fig. 5 Probabilities of getting a solution by WMCH and MCH algorithms on Model RB. Each point is averaged over 30 runs, and the maximum number of repeating step is set to $60N$. The parameter p in WMCH 1, WMCH 2, and WMCH 3 are all set to 0.05

steps, step 2 of PRW 1 and step 2 of PRW 2 make no more contributions than step Blind Local Search. So the effect of PRW should be similar to Blind Local Search on Model RB, and PRW hardly works on Model RB.

7 Conclusion

We have studied performances of pure random walk (PRW) algorithms on a model of random constraint satisfaction problem with growing domains called Model RB. The same threshold behaviors of PRW are shown on Model RB, just like that of Walksat on k-SAT.

From our results, we find that PRW algorithms are more suitable for k-SAT than for Model RB. Taking 3-SAT as an example, Walksat can solve 3-SAT until clause density 2.7, which is not small relative to its satisfiability threshold value of 4.26. But for Model RB, PRW can work until $\frac{1-p}{p}\frac{1}{k\ln N}$, which is very small (tending to 0) relative to its satisfiability threshold value of $-\frac{\alpha}{\ln(1-p)}$ (a constant). This may be due to the fact that the instances of Model RB have large domain size, and a large domain size leads to more constraints and more unsat-constraints (for a given random assignment), while PRW algorithms cannot deal with such a situation.

Backtrack-Free algorithm can solve Model RA (almost the same as Model RB) until a positive constant proportion of $-\frac{\alpha}{\ln(1-p)}$, while it can barely solve k-SAT. Therefore, CSPs with large domain size (such as Model RB) and CSPs with small domain size

(such as k-SAT) may have different properties, and different strategies (such as PRW and BTF) may have different effects on them.

Acknowledgments Partially supported by NSFC 61370052 and 61370156.

References

Achlioptas D, Kirousis L, Kranakis E, Krizanc D, Molloy M, Stamatiou Y (1997) Random constraint satisfaction: a more accurate picture. In: Proceedings of CP. pp 107–120

Alekhnovich M, Ben-Sasson E (2006) Linear upper bounds for random walk on small density random 3-cnfs. SIAM J Comput 36(5):1248–1263

Alphonse E, Osmani A (2008) A model to study phase transition and plateaus in relational learning. In: Proceedings of of ILP. pp 6–23

Barthel W, Hartmann AK, Weigt M (2003) Solving satisfiability problems by fluctuations: the dynamics of stochastic local search algorithms. Phys Rev E 67:066104

Broder AZ, Frieze AM, Upfal E (1993) On the satisfiability and maximum satisfiability of random 3-CNF formulas. In: Proceedings of SODA. pp 322–330

Chao M, Franco J (1986) Probabilistic analysis of two heuristics for the 3-satisfiability problem. SIAM J Comput 15(4):1106–1118

Coja-Oghlan A, Frieze A (2012) Analyzing Walksat on random formulas. In: Proceedings of ANALCO. pp 48–55

Coja-Oghlan A, Feige U, Frieze A, Krivelevich M, Vilenchik D (2009) On smoothed k-CNF formulas and the Walksat algorithm. In: Proceedings of SODA. pp 451–460

Fan Y, Shen J (2011) On the phase transitions of random k-constraint satisfaction problems. Artif Intell 175:914–927

Fan Y, Shen J, Xu K (2012) A general model and thresholds for random constraint satisfaction problems. Artif Intell 193:1–17

Gao Y, Culberson J (2007) Consistency and random constraint satisfaction problems. J Artif Intell Res 28:517–557

Gent I, Macintype E, Prosser P, Smith B, Walsh T (2001) Random constraint satisfaction: flaws and structure. Constraints 6(4):345–372

Huang P, Yin MH (2014) An upper (lower) bound for Max (Min) CSP. Sci China Inf Sci 57:072109

Jiang W, Liu T, Ren T, Xu K (2011) Two hardness results on feedback vertex sets. In: Proceedings of FAW-AAIM. pp 233–243

Kamath A, Motwani R, Palem K, Spirakis P (1995) Tail bounds for occupancy and the satisfiability threshold conjecture. Random Struct Algorithm 7:59–80

Lecoutre C (2009) Constraint networks: techniques and algorithms. Wiley, Hoboken

Liu T, Lin X, Wang C, Su K, Xu K (2011) Large hinge width on sparse random hypergraphs. In: Proceedings of IJCAI. pp 611–616

Liu T, Wang C, Xu K (2014) Large hypertree width for sparse random hypergraphs. J Comb Optim. doi:10.1007/s10878-013-9704-y

Richter S, Helmert M, Gretton C (2007) A stochastic local search approach to vertex cover. In: Proceedings of KI. pp 412–426

Rossi F, Van Beek P, Walsh T (eds) (2006) Handbook of constraint programming. Elsevier, Amsterdam

Schöning U (2002) A probabilistic algorithm for k-SAT based on limited local search and restart. Algorithmica 32:615–623

Schöning U (1999) A probabilistic algorithm for k-SAT and constraint satisfaction problems. In: Proceedings of FOCS. pp 410–414

Semerjian G, Monasson R (2004) A study of pure random walk on random satisfiability problems with physical methods. In: Proceedings of SAT. pp 120–134

Semerjian G, Monasson R (2003) Relaxation and metastability in the random walk SAT search procedure. Phys Rev E 67:066103

Shen J, Ren Y (2014) Bounding the scaling window of random constraint satisfaction problems. J Comb Optim. doi:10.1007/s10878-014-9789-y

Smith BM (2001) Constructing an asymptotic phase transition in random binary constraint satisfaction problems. Theor Comput Sci 265:265–283
Smith BM, Dyer ME (1996) Locating the phase transition in binary constraint satisfaction problems. Artif Intell 81:155–181
Wang C, Liu T, Cui P, Xu K (2011) A note on treewidth in random graphs. In: Proceedings of COCOA. pp 491–499
Xu K, Li W (2000) Exact phase transitions in random constraint satisfaction problems. J Artif Intell Res 12:93–103
Xu K, Li W (2006) Many hard examples in exact phase transitions. Theor Comput Sci 355:291–302
Xu K, Boussemart F, Hemery F, Lecoutre C (2007) Random constraint satisfaction: easy generation of hard (satisfiable) instances. Artif Intell 171:514–534
Xu W (2014) An analysis of backtrack-free algorithm on a constraint satisfaction problem with growing domains (in Chineses). Acta Math Appl Sin (Chin Ser) 37(3):385–392
Zhao C, Zheng Z (2011) Threshold behaviors of a random constraint satisfaction problem with exact phase transitions. Inf Process Lett 111:985–988
Zhao C, Zhang P, Zheng Z, Xu K (2012) Analytical and belief-propagation studies of random constraint satisfaction problems with growing domains. Phys Rev E 85:016106
Zhou G, Gao Z, Liu J (2014) On the constraint length of random k-CSP. J Comb Optim. doi:10.1007/s10878-014-9731-3