Determination of the exponent γ for SAWs on the two-dimensional Manhattan lattice

1999 J. Phys. A: Math. Gen. 32 2931

(http://iopscience.iop.org/0305-4470/32/16/004)

J. Phys. A: Math. Gen. **32** (1999) 2931–2948. Printed in the UK PII: S0305-4470(99)00340-6

Determination of the exponent γ for SAWs on the two-dimensional Manhattan lattice

Sergio Caracciolo†‖, Maria Serena Causo‡¶, Peter Grassberger‡[+] and Andrea Pelissetto§*

† Scuola Normale Superiore and INFN—Sezione di Pisa, I-56100 Pisa, Italy
‡ John von Neumann-Institut für Computing (NIC), Forschungszentrum Jülich, D-52425 Jülich, Germany
§ Dipartimento di Fisica and INFN—Sezione di Roma I, Università degli Studi di Roma 'La Sapienza', I-00185 Roma, Italy

Received 21 December 1998

Abstract. We present a high-statistics Monte Carlo determination of the exponent γ for self-avoiding walks on a Manhattan lattice in two dimensions. A conservative estimate is $\gamma \gtrsim 1.3425 \pm 0.0003$, in agreement with the universal value $\frac{43}{32}$ on regular lattices, but in conflict with predictions from conformal field theory and with a recent estimate from exact enumerations. We find strong corrections to scaling that seem to indicate the presence of a non-analytic exponent $\Delta < 1$. If we assume $\Delta = \frac{11}{16}$ we find $\gamma = 1.3436 \pm 0.0003$, where the error is purely statistical.

1. Introduction

The self-avoiding walk (SAW) is a model which describes the universal properties of flexible chain polymers in a good solvent in the dilute regime. A simple but intriguing modification has been recently introduced to study polymers with an intrinsic orientation [1,2]. This orientation could be due to the presence of dipole moments on the monomers of the chain or to an ordering in the sequence of monomer constituents. On the lattice one considers SAWs with a short-range interaction between different steps of the walk according to their relative orientation. The partition function is simply

$$z_N = \sum_{\{\omega\}} \mathrm{e}^{\beta_a m_a + \beta_p m_p} \tag{1.1}$$

where m_p and m_a are respectively the number of parallel and antiparallel interactions and the sum extends over all SAWs of length N.

This model has a rich phase diagram [3–7] and can be analysed theoretically [1, 2] by mapping it into a complex O(n) model in the limit $n \to 0$. In two dimensions, the theory with a repulsive interaction between parallel bonds, i.e. with $\beta_a = 0$, $\beta_p < 0$, was analysed using conformal invariance techniques [2]. It was shown that the new interaction is truly marginal, giving rise to a line of fixed points. The main consequence is that the partition-function exponent γ should vary continuously with the strength of the orientation-dependent

‖ E-mail address: Sergio.Caracciolo@sns.it
¶ E-mail address: M.S.Causo@fz-juelich.de
[+] E-mail address: P.Grassberger@fz-juelich.de
* E-mail address: pelisset@ibmth.df.unipi.it

0305-4470/99/162931+18$19.50

interaction. The exponent γ is defined from the asymptotic behaviour of the partition function z_N, which should scale asymptotically as

$$z_N = A\mu^N N^{\gamma-1}. \tag{1.2}$$

Here A and μ are non-universal constants, while γ is an exponent that, in the absence of the orientation-dependent interaction, is expected to be universal: it should not depend on the details of the interaction and it should assume the same value for any two-dimensional regular lattice. Using Coulomb-gas techniques, Nienhuis [8] predicted $\gamma_{\rm reg} = \frac{43}{32}$, a value that has been confirmed to high-precision by many numerical computations, see e.g. [9]. On the other hand, for interacting oriented SAWs with $\beta = 0$, $\beta_p < 0$, γ should be a function of β_p. Unfortunately, conformal field theory does not provide definite numerical estimates, although it predicts that $\gamma(\beta_p)$ should decrease monotonically as $\beta_p \to -\infty$.

The square-lattice model was studied by exact enumerations in [3] and by transfer-matrix techniques in [10] finding a very tiny dependence of γ on β_p, thereby supporting the field-theory analysis. When parallel interactions are forbidden, i.e. for $\beta_p = -\infty$, using unbiased approximants it was found [3] that

$$0.006 \lesssim \gamma_{\rm reg} - \gamma \lesssim 0.013. \tag{1.3}$$

The systematic uncertainty due to the extrapolations of two different series is taken into account in this range. The evidence for a non-universal behaviour is not overwhelming, also keeping into account that several problems affected the results of the analysis. As the authors report, they found small shifts of the critical fugacity μ_c from the value it assumes for the ordinary SAW, in contrast with the theoretical result that μ_c should not depend on β_p. More worryingly, an analysis using biased differential approximants with fixed critical fugacity gave a smaller prediction for $\gamma_{\rm reg} - \gamma$, although with much less confidence since most of the approximants were defective. The analysis of Koo [10], based on strips of width $\leqslant 8$, was less precise. For the largest value of β_p that was analysed, $\beta_p = -3$, he finds $\gamma_{\rm reg} - \gamma = 0.018 \pm 0.012$, a difference that is barely significant.

The evidence provided by [3, 10] for a non-universal behaviour of γ was not conclusive and this spurred many workers to improve the result and/or to investigate the problem using different methods. The transfer-matrix analysis was improved in [5]: using larger strips they did not find any evidence of a non-universal behaviour and interpreted previous results as due to short-series effects and to the small size of the strips. Other field-theoretical predictions were also tested. The mean value of m_p for ordinary SAWs was computed by Monte Carlo and exact-numeration methods in [4, 11]: it was found that $\langle m_p \rangle$ converges to a constant as $N \to \infty$, in contrast with the field-theoretical prediction $\langle m_p \rangle \sim \log N$. Recently, the behaviour of $\langle m_p \rangle$ on a cylinder was determined by a Monte Carlo simulation [12], also finding in this case a result in disagreement with the field-theory predictions.

Recently, it was shown [13] that Cardy's original argument implies that the exponent γ for the Manhattan lattice should also be different from the exponent $\gamma_{\rm reg}$. Indeed on this lattice a SAW is oriented by default and parallel interactions are automatically suppressed. From the analysis of long exact-enumeration series the authors of [13] report

$$\gamma_{\rm reg} - \gamma = 0.0053 \pm 0.0030. \tag{1.4}$$

The effect is extremely small. It differs from zero by less than two error bars. The evidence for $\gamma \neq \gamma_{reg}$ is therefore not overwhelmingly persuasive, and the theoretical importance of the problem requires further investigations. Indeed one can suspect that the small deviation is simply a systematic effect due to the corrections to scaling that are not completely taken into account by the analysis.

We have therefore decided to investigate the problem by means of a Monte Carlo simulation, computing the exponent γ for SAWs on a Manhattan lattice. The advantage is that we are able to work with very long walks ($N \leqslant N_{\max} = 32\,000$) and therefore to reduce the unknown systematic uncertainty due to the extrapolation $N \to \infty$.

Our simulations were performed with two different algorithms. The first one, the *join-and-cut* algorithm [14], is a dynamic Monte Carlo algorithm that works in the ensemble of couples of walks with fixed total length. The algorithm is at present the best one to compute the exponent γ since the autocorrelation time in CPU units scales as $N^{1.6}$, while for other algorithms it behaves no better than $N^{\approx 2}$. The second algorithm is a variant of the pruned-enriched Rosenbluth method (PERM) [15]. This is a growth algorithm. Asymptotically, it is slower than the *join-and-cut* algorithm in generating independent configurations. The computer time to generate an independent configuration scales as N^2. But the constant in front of N^2 is very small, for instance the number of monomer additions needed to obtain one independent configuration was numerically found to increase as $\approx 0.008N^2$ for $N > 10\,000$. Therefore, it is possible that the PERM is the most efficient one even for quite long walks. As we shall discuss, with a clever improvement, the *Markovian anticipation*, the PERM is more efficient in providing estimates of γ than the join-and-cut algorithm as long as the length of the sampled walks is less than 10^4. Only for longer walks is the join-and-cut algorithm faster. Another advantage in the present context is that it also gives directly, together with the estimate of the partition sum for chains of length $N_{\max}$ and without extra cost, all partition sums for shorter chains. These estimates for different N are not independent, but just because of this fact they are particularly useful for estimating γ.

The main sources of systematic errors in our analysis are the corrections to scaling, assumed to be of the form

$$\frac{z_N}{A\mu^N N^{\gamma-1}} \approx 1 + \frac{a}{N^{\Delta}} + \frac{a_1}{N^{\Delta_1}} + \cdots \tag{1.5}$$

with $\Delta < \Delta_1 < \cdots$. All numerical evidence for SAWs on regular lattices (square, honeycomb and triangular) indicates that the leading correction is the analytic one [16–18]. On the other hand Saleur [19] predicted $\Delta = \frac{11}{16}$, a result that was confirmed in numerical work on lattice trails [20,21]†. Why this exponent does not show up for SAWs on regular lattices is completely unclear.

For the Manhattan lattice, it has been shown in [13] that the enumeration data are very well fitted using an ansatz with no non-analytic terms‡ with $\Delta < 1$. On the other hand a naive fit of our Monte Carlo data would indicate just the opposite: the results are well fitted assuming a correction-to-scaling exponent of order $\Delta \approx 0.5$–0.7. Of course one should not take this indication too seriously—our data are not precise enough for a serious attempt to determine Δ—but it is fair to say that $\Delta = \frac{11}{16}$ is our preferred value. If we assume $\Delta = \frac{11}{16}$ we obtain

$$\gamma = 1.3436 \pm 0.0003 \tag{1.6}$$

in very good agreement with $\gamma = \gamma_{\text{reg}} = \frac{43}{32} = 1.343\,75$. We have also tried to analyse our data assuming $\Delta = 1$. The quality of the fit is somewhat worse, although one could think that this is simply due to the additional neglected corrections to scaling§ that are still relevant

† An additional hint to $\Delta = \frac{11}{16}$ is the numerical observation of Barkema and Flesia [4] that the average number of loops of length l forming a parallel contact scales as $\langle m_p \rangle_l \sim l^{-1.65\pm0.05}$. From this they estimate that the total number of parallel contacts behaves as $\langle m_p \rangle = a - b/N^{0.65}$. On the other hand, $\langle m_p \rangle$ is proportional to $\mathrm{d}z_N(T)/\mathrm{d}T$ if one includes an orientation dependent interaction, whence there should be generically a term $\sim N^{-0.65} \approx N^{-11/16}$ in $z_N(T)$.

‡ Inclusion of a term with $\Delta = \frac{11}{16}$ worsens the quality of the extrapolation [22].

§ A similar phenomenon occurs for SAWs on the square lattice [18]. If one analyses the end-to-end distance for short SAWs, with a single correction term, one finds $\Delta \approx 0.80$. Only the inclusion of longer walks and more correction

at the values of N we are working (the determination of Δ depends mainly on the data with $N_{\rm tot} = 2000$, i.e. on walks with $N \lesssim 1000$).

We do not know how reliable the estimate (1.6) is and a serious analysis of the systematic errors is practically impossible. In any case, without explicit assumptions on Δ, we can still obtain the lower bound

$$\gamma \gtrsim 1.3425 \pm 0.0003. \tag{1.7}$$

Although this result is lower than the estimate (1.6), it clearly supports the fact that $\gamma = \gamma_{\rm reg}$. The prediction in [13], $\gamma = 1.3385 \pm 0.003$ is instead clearly excluded.

We should point out that if we only analyse data with small values of N, we would obtain a lower estimate for γ, in close agreement with the result (1.4). This shows the crucial role played in this problem by corrections to scaling and the importance of performing simulations for very large values of N.

2. The join-and-cut algorithm on the Manhattan lattice

2.1. Description of the algorithm

In this section we will define the pivot and the join-and-cut algorithm on a Manhattan lattice. The Manhattan lattice is a two-dimensional square lattice on which bonds are directed in such a way that adjacent rows (columns) have antiparallel directions, corresponding to the traffic pattern in Manhattan. Although bonds are directed, there is no overall directional bias. Explicitly we will assume the following orientations: a vertical bond connecting the points of coordinates (x, y), $(x, y+1)$ is directed upward if x is even, downward if x is odd; a horizontal bond connecting the points of coordinates (x, y) and $(x+1, y)$ is directed to the left if y is even, to the right if y is odd.

Let us now define the pivot algorithm [23–25]. It works in the ensemble of fixed-length walks with free endpoints: we will be interested in self-avoiding walks, but the algorithm can also be applied in a very efficient way to the Domb–Joyce model [26, 27], to power-law walks [26], and to interacting polymers far from the Θ-transition [28, 29].

The algorithm works as follows [25]. Given an N-step SAW ω starting at the origin and ending anywhere, $\omega \equiv \{\omega(0), \ldots, \omega(N)\}$, an iteration of the algorithm consists of the following steps:

(i) Choose randomly, with uniform probability, an integer $k \in \{0, 1, \ldots, N-1\}$.
(ii) Choose with probability $P(g)$ an element g of the lattice point symmetry group. The probability must satisfy $P(g) = P(g^{-1})$ to ensure detailed balance.
(iii) Propose a pivot move $\omega \to \omega'$ defined by

$$\omega'(i) = \begin{cases} \omega(i) & \text{for} \quad 1 \leqslant i \leqslant k \\ \omega(k) + g(\omega(i) - \omega(k)) & \text{for} \quad k+1 \leqslant i \leqslant N. \end{cases} \tag{2.1}$$

The proposed move is accepted if ω' is self avoiding; otherwise it is rejected and we stay with ω.

The probability $P(g)$ must be such to ensure ergodicity: as discussed in [25, 30] not all symmetry transformations are needed. In particular [30] the pivot algorithm is ergodic on a square lattice if $P(g)$ is non-vanishing only for diagonal reflections, that is for reflections through lines of slope ± 1.

terms with $\Delta_i > 1$ gives $\Delta \approx 1.0$.

For the Manhattan lattice the point symmetry group is much smaller than the symmetry group of the square lattice. Indeed the lattice is symmetric only with respect to lines of slope -1 going through lattice points (x, y) with $x + y$ even, and with respect to lines of slope $+1$ going through (x, y) with $x + y$ odd. Therefore we will modify step (ii) in the following way:

(ii) if $\omega_x(k) + \omega_y(k)$ is even (resp. odd), let g be the reflection with respect to a line of slope -1 (resp. $+1$), going through $\omega(k)$.

Since $g^2 = 1$, detailed balance is automatically satisfied. The tricky point is ergodicity. It can be proved by slightly modifying the proof of section 5 in [30]. Using the same definitions, the basic observation is the following: if $\omega(k) \equiv (x_k, y_k)$ is a walk site belonging to a diagonal support line of slope -1 (resp. $+1$), then $x_k + y_k$ is even (resp. odd). Using this fact the proof works without any change.

We wish now to define the join-and-cut algorithm [14]. The tricky point here is that in general, given two walks on the Manhattan lattice, their concatenation is a walk that does not respect the bond orientation of the lattice. More precisely, given two walks ω_1, ω_2 of lengths N_1 and N_2, the concatenated walk $\omega = \omega_1 \circ \omega_2$ respects the orientation of the lattice only if†

$$\mathrm{mod}(\omega_{1x}(N_1) - \omega_{2x}(0), 2) = 0 \qquad \mathrm{mod}(\omega_{1y}(N_1) - \omega_{2y}(0), 2) = 0. \quad (2.2)$$

This reflects the fact that only translations of two steps in each direction are symmetries of the lattice. A second consequence of the bond orientation of the Manhattan lattice is the following: consider a walk ω and cut it into two parts ω_1 and ω_2 such that $\omega = \omega_1 \circ \omega_2$. In general there is no lattice translation T such that the translated walk $T\omega_2$ respects the bond orientation of the lattice and satisfies $(T\omega_2)(0) = \omega_1(0) = \omega(0)$. Indeed this happens only if

$$\mathrm{mod}(\omega_{1x}(0) - \omega_{2x}(0), 2) = 0 \qquad \mathrm{mod}(\omega_{1y}(0) - \omega_{2y}(0), 2) = 0. \quad (2.3)$$

With these two observations in mind we define our ensemble in the following way: $T_{N_{\mathrm{tot}}}$ consists of all pairs (ω_1, ω_2) of SAWs, each walk starting either in $(0, 0)$ or in $(1, 1)$ and ending anywhere, such that the *total* number of steps in the two walks is some fixed *even* number N_{tot}. Moreover we require the lengths of ω_1 and ω_2 to be even. Explicitly

$$T_{N_{\mathrm{tot}}} = \bigcup_{k=1}^{N_{\mathrm{tot}}/2-1} (S_{2k}(0, 0) \cup S_{2k}(1, 1)) \times (S_{N_{\mathrm{tot}}-2k}(0, 0) \cup S_{N_{\mathrm{tot}}-2k}(1, 1)) \quad (2.4)$$

where $S_k(x, y)$ is the set of k-step walks starting from (x, y) and ending anywhere. Each pair in the ensemble is given equal weight: therefore the two walks are not interacting except for the constraint on the sum of their lengths. Notice that there is a one-to-one correspondence between $S_k(0, 0)$ and $S_k(1, 1)$ so that the ensemble defined in eqaution (2.4) is equivalent to

$$T_{N_{\mathrm{tot}}} = \bigcup_{k=1}^{N_{\mathrm{tot}}/2-1} S_{2k}(0, 0) \times S_{N_{\mathrm{tot}}-2k}(0, 0). \quad (2.5)$$

One sweep of the algorithm consists of the following steps:

(i) Starting from a pair of walks (ω_1, ω_2), we update each of them independently using some ergodic fixed-length algorithm. We use the pivot algorithm we described above.

(ii) With probability $\frac{1}{2}$ we interchange ω_1 and ω_2, i.e. $(\omega_1, \omega_2) \to (\omega_2, \omega_1)$.

(iii) We first check the parity of the endpoints: if $\mathrm{mod}(\omega_{1x}(N_1) - \omega_{2x}(0), 2) = 1$, we stay with (ω_1, ω_2). Otherwise we attempt a *join-and-cut* move. We choose with uniform probability $k \in \{1, \ldots, N_{\mathrm{tot}}/2 - 1\}$. Then we concatenate the two walks ω_1 and ω_2 forming a new (not necessarily self-avoiding) walk $\omega_{\mathrm{conc}} = \omega_1 \circ \omega_2$; then we cut ω_{conc} creating two new walks ω'_1 and ω'_2 of lengths $2k$ and $N_{\mathrm{tot}} - 2k$. If ω'_1 and ω'_2 are self-avoiding we keep them; otherwise the move is rejected and we stay with ω_1 and ω_2.

† The only exception to this rule is when ω_2 is a rod.

It is easy to see that the full algorithm is ergodic.

The algorithm defined on the Manhattan lattice works essentially in the same way as the standard one. The only important difference is that one must check the parities of $\omega_1(N_1)$ and $\omega_2(0)$ before attempting the join-and-cut move. This check is successful in 50% of the cases and thus the algorithm we have defined above should be essentially equivalent to the standard algorithm in which one performs two pivot updates of each walk for every join-and-cut move (in the notation of [14] it corresponds to the algorithm with $n_{\rm piv} = 2$). In principle it is easy to avoid this 50% rejection by modifying the third step of the algorithm in the following way:

(iii) *(improved join-and-cut move).* If $\mathrm{mod}(\omega_{1x}(N_1) - \omega_{2x}(0), 2) = 1$, let $\omega_{\rm conc} = \omega_1 \circ R\omega_2$, where $R\omega_2$ is the walk reflected with respect to the line of slope +1 going through $\omega_2(0)$; if $\mathrm{mod}(\omega_{1x}(N_1) - \omega_{2x}(0), 2) = 0$, let $\omega_{\rm conc} = \omega_1 \circ \omega_2$. Then proceed as before.

This modification attempts twice the number of join-and-cut moves with respect to the previous one, and thus it should be more efficient. However, the difference in performance is not expected to be large. Indeed from the analysis of the autocorrelation times in [14], we obtain† that the ratio $\tau_{\rm int}(n_{\rm piv} = 2)/\tau_{\rm int}(n_{\rm piv} = 1)$ is equal to 1.6, 1.5, 1.2 for $N_{\rm tot} = 500, 2000, 8000$ respectively. For the algorithm on the Manhattan lattice we expect $\tau_{\rm int}(\mathrm{non-impr})/\tau_{\rm int}(\mathrm{imp})$ to be very similar to the previous ratio: therefore for $N_{\rm tot} = 8000$ we expect that the autocorrelation time for the original algorithm to be only 20–30% higher than the corresponding time for the improved one. One should also keep in mind that one iteration of the improved algorithm is slower than one iteration of the unimproved one, since one is doing twice the number of join-and-cut attempts. Therefore we expect that for $N_{\rm tot} = 8000$ the improved algorithm is only 10–20% better than the original one. Since the implementation of the improved algorithm required major changes of our codes, we decided to work with the unimproved version we described above.

2.2. Determination of γ from join-and-cut data

Let us now discuss how the critical exponent γ can be estimated from the Monte Carlo data produced by the join-and-cut algorithm.

Let us start by noticing that the random variable N_1, the length of the first walk, has the distribution

$$\overline{\pi}(N_1) = \begin{cases} \dfrac{z_{N_1} z_{N_{\rm tot}-N_1}}{Z(N_{\rm tot})} & \text{if} \quad N_1 \text{ is even} \\ 0 & \text{if} \quad N_1 \text{ is odd} \end{cases} \tag{2.6}$$

for $1 \leqslant N_1 \leqslant N_{\rm tot} - 1$; here $Z(N_{\rm tot})$ is the obvious normalization factor and z_N is the number of N-step walks going from the origin to any lattice point whose asymptotic behaviour for large N is given by (1.2). The idea is then to make inferences of γ from the observed statistics of N_1. Of course the problem is that (1.2) is an asymptotic formula valid only in the large-N regime. We will thus proceed in the following way: we will suppose that (1.2) is valid for all $N \geqslant N_{\rm min}$ for many increasing values of $N_{\rm min}$ and correspondingly we will get estimates $\hat{\gamma}(N_{\rm tot}, N_{\rm min})$; these quantities are *effective* exponents that depend on $N_{\rm min}$ and that give correct estimates of γ as $N_{\rm min}$ and $N_{\rm tot}$ go to infinity.

The determination of γ from the data is obtained using the maximum-likelihood method (see e.g. [31]). We will present here only the results: for a detailed discussion we refer the reader to [14].

† Notice that in [14] $\tau_{\rm int}(n_{\rm piv})$ is reported in units of two iterations, each iteration consisting of $n_{\rm piv}$ pivot attempts and one join-and-cut attempt. In order to make a correct comparison we should multiply $\tau_{\rm int}(n_{\rm piv})$ by $n_{\rm piv}$. Thus what we report as $\tau_{\rm int}(n_{\rm piv} = 2)$ corresponds to $2\tau_{\rm int}(n_{\rm piv} = 2)$ of [14].

Table 1. Number of iterations and CPU times per iteration for various values of $N_{\rm tot}$. CPU-times are expressed in ms and refer to an Alpha-Station 600 Mod 5/266.

$N_{\rm tot}$	$N_{\rm iter}$	CPU-time
500	10^9	0.248(1)
2000	12×10^8	0.846(8)
8000	33×10^8	2.41(3)
32 000	23×10^8	9.08(2)

Given $N_{\rm min}$, consider the function (from now on we suppress the dependence on $N_{\rm tot}$)

$$\theta_{N_{\rm min}}(N_1) = \begin{cases} 1 & \text{if } N_{\rm min} \leqslant N_1 \leqslant N_{\rm tot} - N_{\rm min} \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

and let X be the random variable

$$X = \log[N_1(N_{\rm tot} - N_1)]. \tag{2.8}$$

Then define

$$X^{\rm cens}(N_{\rm min}) = \frac{\langle X\theta_{N_{\rm min}}\rangle}{\langle\theta_{N_{\rm min}}\rangle} \tag{2.9}$$

where the average $\langle\,\cdot\,\rangle$ is taken in the ensemble $T_{N_{\rm tot}}$ sampled by the join-and-cut algorithm. The quantity defined in equation (2.9) is estimated in the usual way from the Monte Carlo data obtaining in this way $X^{\rm cens}_{\rm MC}(N_{\rm min})$. Then $\hat\gamma(N_{\rm min})$ is computed by solving the equation

$$X^{\rm cens}_{\rm MC}(N_{\rm min}) = [X]_{{\rm th},\hat\gamma}(N_{\rm min}) \tag{2.10}$$

where, for every function of N_1, we define

$$[f(N_1)]_{{\rm th},\gamma}(N_{\rm min}) \equiv \frac{\sum_{N_1=N_{\rm min}}^{N_{\rm tot}-N_{\rm min}} f(N_1)N_1^{\gamma-1}(N_{\rm tot}-N_1)^{\gamma-1}}{\sum_{N_1=N_{\rm min}}^{N_{\rm tot}-N_{\rm min}} N_1^{\gamma-1}(N_{\rm tot}-N_1)^{\gamma-1}} \tag{2.11}$$

here the sum is extended over *even* values of N_1. The variance of $\hat\gamma(N_{\rm min})$ is then given by

$$\text{Var}\,[\hat\gamma(N_{\rm min})] = \frac{\text{Var}\,(X^{\rm cens}_{\rm MC}(N_{\rm min}))}{([X;X]_{{\rm th},\hat\gamma}(N_{\rm min}))^2} \tag{2.12}$$

where $[X;X] = [X^2] - [X]^2$. We must finally compute Var $(X^{\rm cens}_{\rm MC}(N_{\rm min}))$. As this quantity is defined as the ratio of two mean values (see equation (2.9)) one must take into account the correlation between denominator and numerator. Here we have used the standard formula for the variance of a ratio (valid in the large-sample limit)

$$\text{Var}\left(\frac{A}{B}\right) = \frac{\langle A\rangle^2}{\langle B\rangle^2}\text{Var}\left(\frac{A}{\langle A\rangle} - \frac{B}{\langle B\rangle}\right). \tag{2.13}$$

2.3. Dynamic behaviour

Let us first discuss the simulations using the join-and-cut algorithm. We have performed high-statistics runs at $N_{\rm tot} = 500, 2000, 8000$ and 32 000. In table 1 we report the number of iterations and the CPU time per iteration on an Alpha-Station 600 Mod 5/266. The total join-and-cut runs took 1 year of CPU on this machine.

In table 2 we report the acceptance fraction of the pivot and of the join-and-cut moves, and, for two different values of $N_{\rm min}$, the autocorrelation times for the observable

$$Y(N_{\rm min}) = \frac{X\theta_{N_{\rm min}}}{\langle X\theta_{N_{\rm min}}\rangle} - \frac{\theta_{N_{\rm min}}}{\langle\theta_{N_{\rm min}}\rangle} \tag{2.14}$$

Table 2. Acceptance fraction for the pivot move ($f_{\rm piv}$), for the join-and-cut move ($f_{\rm jc}$) and autocorrelation times for the various values of $N_{\rm tot}$. Autocorrelation times are expressed in units of two iterations.

$N_{\rm tot}$	$f_{\rm piv}$	$f_{\rm jc}$	$\tau_{{\rm int},Y(1)}$	$\tau_{{\rm int},Y(100)}$
500	0.451 943(11)	0.155 468(11)	4.912 ± 0.031	8.171 ± 0.023
2000	0.344 631 6(86)	0.099 862 9(77)	14.30 ± 0.23	19.38 ± 0.12
8000	0.262 405 0(83)	0.062 992 5(65)	49.6 ± 1.2	55.8 ± 1.0
32 000	0.199 693(46)	0.039 330(31)	202.4 ± 7.4	206.7 ± 7.0

that, according to equation (2.13), controls the errors on γ. Notice that $Y(2) = X/\langle X\rangle - 1$, so that $\tau_{{\rm int},Y(2)} = \tau_{{\rm int},X}$.

To compute the autocorrelation times we used the recipe of [32, appendix C]. Indeed the autocorrelation function has a very long tail with a very small amplitude, due to the fact that $\tau_{\rm exp} \gg \tau_{{\rm int},Y}$. For this reason the self-consistent windowing method in [25, appendix C] does not work correctly and underestimates the autocorrelation time. Following [32], we have computed $\tau_{{\rm int},Y}$ as

$$\tau_{{\rm int},Y} = \bar{\tau}_{{\rm int},Y} + \tau_{{\rm tail},Y}. \tag{2.15}$$

Here $\bar{\tau}_{{\rm int},Y}$ is the autocorrelation time computed using the self-consistent windowing method of [25, appendix C] with window $W = 15\bar{\tau}_{{\rm int},Y}$ and

$$\tau_{{\rm tail},Y} = \rho_Y(W) W \log\left(\frac{N}{W f_{\rm piv}}\right) \tag{2.16}$$

where $\rho_Y(t)$ is the normalized autocorrelation function and $f_{\rm piv}$ is the acceptance fraction of the pivot algorithm. Of course equation (2.16) is a very rough ad hoc estimate of the contribution of the tail. For X, it amounts to approximately 35% for $N_{\rm tot} = 32\,000$, 24% for $N_{\rm tot} = 8000$, 15% for $N_{\rm tot} = 2000$, and 5% for $N_{\rm tot} = 500$. Based on our experience with the pivot algorithm we have assigned to $\tau_{{\rm tail},Y}$ an error of 10%. The error on $\bar{\tau}_{{\rm int},Y}$ was instead computed as in [25].

We have first analysed the acceptance fractions. The general analysis in [14] predicts $f_{\rm piv} \sim N^{-p}$ and $f_{\rm jc} \sim N^{-q}$, with $p \approx 0.19$ and $q \approx \gamma - 1 \approx 0.34$. A least-square fit to the data of table 2 gives

$$p = 0.196\,05 \pm 0.000\,01 \qquad \chi^2_1 = 603 \tag{2.17}$$

$$q = 0.325\,22 \pm 0.000\,04 \qquad \chi^2_1 = 9197. \tag{2.18}$$

The very large value of χ^2 indicates that the quoted errors on p and q, that are of a purely statistical nature, are unreliable. Indeed a simple power law does not fit the data at this level of precision. A more realistic estimate of the errors would be

$$p = 0.1960 \pm 0.0002 \qquad q = 0.325 \pm 0.004. \tag{2.19}$$

We have then performed an analogous analysis for the autocorrelation time $\tau_{{\rm int},Y(N_{\rm min})}$. A least-square fit of the form

$$\tau_{{\rm int},X} \sim N^z \tag{2.20}$$

gives

$$z = \begin{cases} 0.847 \pm 0.006 & \chi^2_1 = 71 \quad \text{for} \quad N_{\rm min} = 1 \\ 0.672 \pm 0.004 & \chi^2_1 = 280 \quad \text{for} \quad N_{\rm min} = 100 \\ 0.747 \pm 0.003 & \chi^2_1 = 50 \quad \text{for} \quad N_{\rm min} = 200. \end{cases} \tag{2.21}$$

Again, the very large value of χ^2 indicates that the error bars are underestimated by at least a factor of ten. The results are somewhat higher than the estimate reported in [14],

$z = 0.70 \pm 0.03$. This is evident if one directly compares our results of table 2 with the values of $\tau_{\rm int}$ reported in [14] (see footnote on p 6): $\tau_{\rm int} = 4.38(6)$, 9.47(21), 20.0(6) for $N_{\rm tot} = 500$, 2000 and 8000, respectively. For $N_{\rm tot} = 500$ the autocorrelation times are similar, while for $N_{\rm tot} = 8000$ there is a factor-of-two difference. It is very difficult to believe that the join-and-cut algorithm has a different dynamic behaviour on the square lattice and on the Manhattan lattice. We have thus tried to understand if the authors in [14] underestimated the autocorrelation times. Therefore we have determined $\tau_{\rm int}$ from our data using the self-consistent windowing procedure with a window of $5\tau_{{\rm int},X}$ as done in [14]. We obtain $\tau_{\rm int} = 4.166(2)$, 10.093(6), 29.13(2), 98.0(2) for $N_{\rm tot} = 500$, 2000, 8000 and 32 000, respectively. A fit of these results gives $z = 0.7271(2)$. This is now in good agreement with the results of [14].

In conclusion we believe a fair estimate would be

$$z = 0.75 \pm 0.05. \tag{2.22}$$

As noticed in [14], with a clever implementation of the algorithm, it is possible that the CPU-time per iteration increases as N^σ with $\sigma < 1$. In particular it was predicted that $\sigma = 1-p$ where p is the exponent that controls the pivot acceptance fraction. In two dimensions we expect $p = 0.1953(21)$ [25] and this is confirmed by our data on the acceptance fraction. From the data of table 2 we obtain

$$\sigma = 0.865(1) \qquad \chi^2_1 = 55. \tag{2.23}$$

Keeping into account that the error bars are underestimated, we find reasonable agreement with the prediction $\sigma = 1 - p$. Of course what one is really interested in is the autocorrelation times expressed in CPU units.

We find

$$\tau_{\rm int} \sim N^{z+\sigma} \sim N^{1.6}. \tag{2.24}$$

The algorithm is not optimal, but still represents an improvement with respect to other algorithms that behave as $N^{\approx 2}$.

3. PERM

3.1. Description of the algorithm

The pruned-enriched Rosenbluth method (PERM) is a chain-growth algorithm based on the well known Rosenbluth (inverse restricted sampling) [33] algorithm. In the latter, chains are grown step by step. In unbiased sampling, steps are randomly generated, and, if a new step violates the self-avoidance constraint, the configuration is discarded and one restarts from scratch. This leads to exponential 'attrition', i.e. the number of attempts needed to generate a chain of length n increases exponentially with n.

In Rosenbluth sampling, such steps are replaced (if possible) by 'correct' steps. This diminishes the attrition problem, but it leads to a bias that has to be compensated by a weight factor. If m_n is the number of free sites where to place the nth monomer (i.e., the number of allowed steps at time n), then the weight of a chain of length N is $W_N \propto \prod_{n=1}^{N} m_n$. The proportionality factor depends on the ensemble one wants to simulate, e.g. it is constant for the canonical ensemble with fixed fugacity. The main problem with the Rosenbluth method is the fact that these weights show large fluctuations for large N. Thus even large samples can be completely dominated by just a few events, which leads to large statistical errors. Even worse, for very large N, the part of the distribution of weights which should dominate might not have been sampled at all, and this can lead to systematic errors. More precisely, while the method

is guaranteed to give an unbiased estimate of the partition sum, due to these fluctuations and to the convexity of the logarithm, estimates of free energies are systematically too small.

The main idea in PERM [15] is to watch the weight W_N as N is growing, and to compare it with the estimated average weight for this value of N. If W_N is judged too big, a clone of the present configuration is made, both clones are attributed a weight $W_N/2$, and both are independently continued ('enrichment'). On the other hand, if W_N is too small, a (pseudo-) random number r is drawn uniformly from $[0, 1]$. If $r < \frac{1}{2}$, the configuration is abandoned ('pruning'), otherwise it is kept and its weight is doubled. This is most easily implemented by recursive function calls. A pseudocode is given in [15].

In this way the problem of large weight fluctuations is avoided, although not entirely. The set of generated configurations is now correlated due to cloning. Let us call a *tour* a set of configurations obtained by cloning from the same 'ancestor'. Although the weights of individual configurations vary between narrow limits, the weights of tours can fluctuate very much. If so, we are basically back at the same problem. This can be avoided by making additional biases. For instance, before each step one might look ahead k steps, and choose the direction accordingly [34]. This is rather time consuming for large k, nevertheless it can be efficient for dense (collapsed) systems.

For diluted systems, as it is the case here, the efficiency can be improved using *Markovian anticipation* [12, 35]. Here one keeps the last k steps in memory, and biases the next step on the basis of the statistics of $(k + 1)$-step configurations accumulated during a previous run.

On the Manhattan lattice, an N-step walk with fixed starting point and fixed direction of the first step can be encoded as a binary sequence of length $N - 1$. A straight step is encoded as '0', while a bend by $\pm 90^\circ$ is encoded as '1'. Notice that one does not have to specify the direction of the bend, as only one bend is allowed at any time step. If the starting point has even x and y and the first step is upward, then the first bend is to the right (left) if it occurs at even (odd) times. After that, each bend is in the same (opposite) direction as the previous one if the number of in-between straight steps is even (odd). Let $\boldsymbol{S} = (s_{-k}, \ldots, s_0) \equiv (\boldsymbol{s}, s_0)$ denote a binary string of length $k + 1$, and let $\mathcal{C}_{N,m}(\boldsymbol{S})$ be the *cylinder set* of all N-step walks starting upward from $(x, y) = (0, 0)$, for which steps $m - k, \ldots, m$ are given by $\boldsymbol{S}$. The total weight of this set in an unbiased ensemble is denoted by $W_{N,m}(\boldsymbol{S})$. Finally, we consider

$$p_{N,m}(s_0|\boldsymbol{s}) = \frac{W_{N,m}(\boldsymbol{S})}{\sum_{s_0'=0,1} W_{N,m}(\boldsymbol{s}, s_0')}. \tag{3.1}$$

If SAWs were a kth order Markov process, we would obtain perfect importance sampling for N-step walks if we would select the mth step according to $p_{N,m}(s_0|\boldsymbol{s})$. Given the fact that SAW's are not Markovian, equation (3.1) comes closest to importance sampling given a finite (k-step) memory.

In practice, equation (3.1) is not applicable as it stands, since it is practically impossible to acquire and store the necessary statistics for all m and N. First of all we use the fact that $p_{N,m}(s_0|\boldsymbol{s})$ becomes independent of N and m in the limit $m, N - m \gg k$. We therefore estimate it for large values of m and N by accumulating statistics for all N larger than some lower threshold (we used $N > 400$), and $N - m$ fixed (we used $N - m = 200$). The fact that this $p(s_0|\boldsymbol{s})$ does not give correct importance sampling for $m \approx N$ is not serious and does not reduce much the efficiency. More serious is the fact that it would be inappropriate for small N and m. In particular, for $m < k$ we have no string $\boldsymbol{s}$ to condition upon. We dealt with this problem by tapering the symbol sequence with a string of k zeros, and by taking step s_0 in the

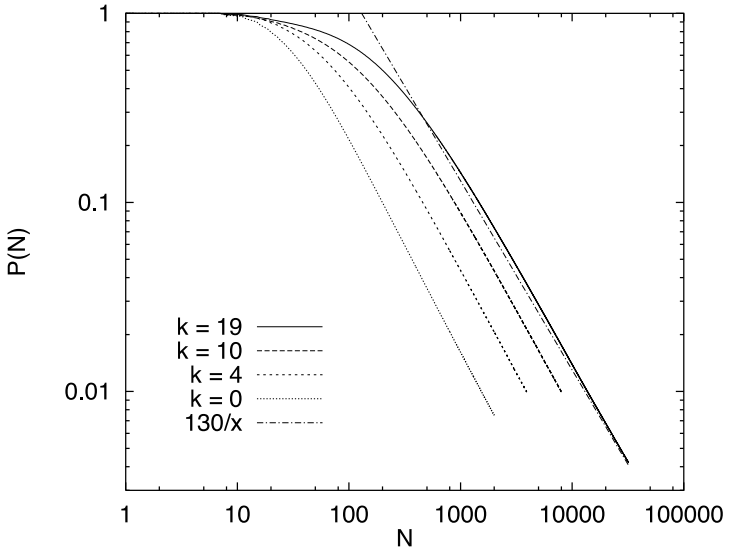

Figure 1. Probabilities to reach length N at least once in a tour, plotted against N. The curves are for PERM with k-step Markovian anticipation, with $k = 19, 10, 5$, and 0. The straight line is $130/N$.

mth step with probability

$$p_m(s_0|\boldsymbol{s}) = \frac{\text{const}/m + \lim_{m,N-m\to\infty} p_{N,m}(s_0|\boldsymbol{s})}{\sum_{s_0'=0,1}[\text{const}/m + \lim_{m,N-m\to\infty} p_{N,m}(s_0|\boldsymbol{s})]} \tag{3.2}$$

with const ≈ 20.

Notice that all this fiddling is not crucial. The method *per se* is correct even if the probabilities for taking the mth step were chosen badly, provided the probability is not set equal to zero for any allowed step. But efficiency can increase substantially with a good choice. Efficiency can be measured in several ways. The most straightforward method consists in measuring tour-to-tour fluctuations. Since subsequent tours are statistically strictly uncorrelated, this gives immediately estimates for the error bars of any measured observable. In practice, not to waste CPU time, we bunched tours in groups of typically 10^4 tours, and measured average values and fluctuations over these bunches.

A less direct but more instructive measure of efficiency is the number of tours in which a length N is reached in at least one configuration, normalized to the number of all tours. For other modified chain-growth algorithms such as incomplete enumeration [36] or the Berretti–Sokal (BS) algorithm [37], it seems that this number decreases as const$/N$, provided the cloning and pruning parameters are adjusted so that the total number of generated walks is independent of N. Improving the details of the algorithm cannot change this scaling law, but can change the constant considerably. For incomplete enumeration and the BS algorithm, the constant is O(1). For PERM without Markovian anticipation const $\approx$ 10. For Markovian anticipation with $k = 19$ we found const ≈ 130, see figure 1.

A last check for efficiency is the following. As we said, the weights of tours can vary substantially, to the point that the distribution of these weights might be not properly sampled in the region which should dominate statistical averages. If this happens, the estimates of the free energies are too low, and all the errors are systematically underestimated. To show that this is not the case in the present simulation, we show in figure 2 the measured distribution of tour

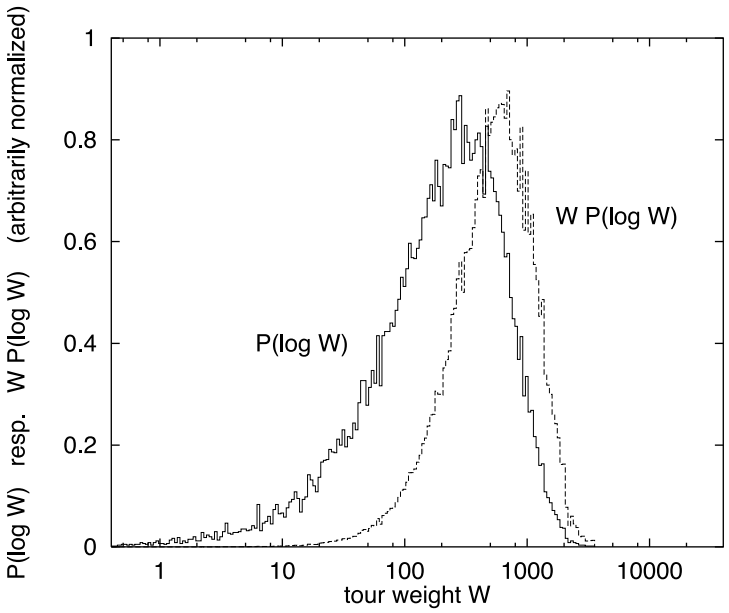

Figure 2. Full line: histogram $P(\log W)$ of tours with fixed weight W, on a logarithmic horizontal scale. Normalization is arbitrary. Broken line: $W \times P(\log W)$, again with arbitrary normalization.

weights for $N = 32\,000$, together with the weighted distribution. We see that the maximum of the weighted distribution occurs at a weight which is still well within our sampling distribution.

3.2. Determination of γ using PERM data

The raw data produced by PERM are estimates of the partition sum z_N, for all $N \leqslant N_{\max}$. It is then straightforward to obtain estimates of γ. Assuming equation (1.2), the exponent γ can be estimated from ratios of z_{N_i} for three different values of N_i. Indeed, if one chooses two positive constants $a < 1 < b$, and defines

$$\gamma_{\text{eff}}(N) = \frac{x \log z_{aN} + y \log z_{bN} - \log z_N}{x \log a + y \log b} \tag{3.3}$$

one checks easily that $\gamma_{\text{eff}}(N) = \gamma$, provided that [38]

$$x + y = 1 \qquad ax + by = 1. \tag{3.4}$$

If equation (1.2) is not fulfilled exactly, equations (3.3) and (3.4) define effective N-dependent exponents. These exponents still depend on a and b. It is clear that statistical errors will be small if $a \ll 1 \ll b$, but then a wide range of chain lengths are needed. Since $N_{\max}$ is fixed and cannot be exceeded, this means that aN will be small, and systematic errors will become large if we wish to reduce statistical errors. In addition to the ratio b/a, one has the product $a \times b$ at one's disposal. For fixed b/a, its optimal value depends on the way how the statistical error on z_N increases with N. In the present case, they increase roughly as $\sqrt{N}$. For practical reasons one also wishes a^{-1} and b to be simple numbers. We found that good results were obtained with $a = \frac{1}{2}$, $b = 4$ (and therefore $x = \frac{6}{7}$, $y = \frac{1}{7}$).

The computation of the error bars is not trivial. Indeed one should take into account the fact that different z_N are correlated and therefore compute the full covariance matrix, which is practically impossible. Therefore we proceeded differently to obtain error estimates. For

Table 3. Number of PERM tours (column 2) with $N \leqslant N_{\max}$, total number of configurations with $N = N_{\max}$ (column 3), number of independent configurations with $N = N_{\max}$ (i.e., of tours which reached $N = N_{\max}$; column 4), and CPU time per independent configuration (expressed in ms, on an Alpha-Station 600 Mod 5/266; column 5).

$N_{\max}$	N_{tour}	N_{config}	N_{indep}	CPU time
500	1.0×10^6	1.140×10^6	2.637×10^5	4.985
2000	8.876×10^8	1.010×10^9	6.533×10^7	70.53
4000	1.756×10^8	2.016×10^8	6.359×10^6	298.6
8000	3.421×10^8	4.035×10^8	6.296×10^6	1272.6(4)
32 000	2.792×10^7	3.703×10^7	1.154×10^5	$244(7) \times 10^2$

each bunch of tours we estimated $[\gamma_{\text{eff}}(N)]_{\text{bunch}}$ according to equation (3.3), and from these values we obtained the variance of the latter. Notice that we did *not* use this procedure to obtain $\gamma_{\text{eff}}(N)$ itself by taking the average over all bunches, as this would introduce a bias.

In addition, in order to improve the estimates of $X^{\text{cens}}(N_{\min})$, cf equation (2.9), obtained using the join-and-cut algorithm, we evaluated this quantity using the PERM estimates of z_N. The estimate of the mean value is trivial. For the error bars, we evaluated $[X^{\text{cens}}(N_{\min})]_{\text{bunch}}$ over bunches of tours and then estimated their variance. We used the same trick as for γ_{eff}. Again, notice that we did not use this procedure to estimate the mean value itself, since this introduces a bias. We have checked, however, that this bias is extremely small.

3.3. Dynamic results

Let us now discuss the simulations using PERM. We have performed high-statistics runs using Markovian anticipation with $k = 19$. The dynamic data of the runs are reported in table 3. The total PERM simulation would have taken approximately 7 months of CPU time on an Alpha-Station 600 Mod 5/266.

We have first of all performed an analysis of the CPU time per independent configuration, which we expect to scale as aN^z with $z \approx 2$. Fitting the data of table 3, and including in each fit only the data with $N \geqslant N_{\min}$ in order to detect systematic effects, we have

$$z \approx \begin{cases} 1.92 & N_{\min} = 500 \\ 2.08 & N_{\min} = 2000 \end{cases} \tag{3.5}$$

$$a \approx \begin{cases} 0.000\,03 & N_{\min} = 500 \\ 0.000\,01 & N_{\min} = 2000. \end{cases} \tag{3.6}$$

The expected exponent $z \approx 2$ is clearly recovered. Let us now compare these results with the join-and-cut estimates. For the latter algorithm, the CPU time per independent configuration scales as (cf tables 1 and 2) $\approx 0.0003N^{1.6}$ and therefore the join-and-cut is faster in generating one independent configuration than the PERM as long as $N \gtrsim 1000$. This may not be, however, a fair comparison of the two algorithms. Indeed what we call an 'independent configuration' for PERM is a bunch of walks which are correlated, but which still contain more information than a single walk. In order to make a fair comparison it is better to consider an observable and compare the CPU time needed to obtain the same error bars. We have therefore compared the results for our preferred observable $X^{\text{cens}}(N_{\min})$, see table 4. We find that for $N_{\text{tot}} = 32\,000$ both algorithms require essentially the same CPU time to produce data with the same error bars. Only for longer walks the join-and-cut algorithm is more efficient. Notice that this result is true if we use the Markovian anticipation. In the absence of this improvement, PERM would

Table 4. Raw data for $X^{\rm cens}_{\rm MC}$. We report separately the results obtained with the PERM and join-and-cut algorithms.

	$N_{\rm tot} = 500$			$N_{\rm tot} = 2000$	
$N_{\rm min}$	$X^{\rm cens}_{\rm PERM}$	$X^{\rm cens}_{\rm jc}$	$N_{\rm min}$	$X^{\rm cens}_{\rm PERM}$	$X^{\rm cens}_{\rm jc}$
2	10.599 798 20(1214)	10.599 773 4(851)	2	13.373 844 1(259)	13.374 033(132)
20	10.682 816 51(924)	10.682 800 3(692)	50	13.428 551 7(215)	13.428 669(115)
40	10.757 165 91(695)	10.757 131 6(555)	100	13.479 698 2(176)	13.479 777(100)
60	10.818 015 10(514)	10.817 977 0(447)	150	13.524 659 2(144)	13.524 727(87)
80	10.868 504 52(365)	10.868 480 7(356)	200	13.564 436 9(118)	13.564 528(77)
100	10.910 576 82(248)	10.910 578 7(280)	250	13.599 809 6(96)	13.599 867(67)
120	10.945 532 80(170)	10.945 560 5(216)	300	13.631 361 0(77)	13.631 408(58)
140	10.974 279 40(116)	10.974 288 9(162)	350	13.659 541 1(61)	13.659 569(51)
160	10.997 466 09(70)	10.997 473 7(115)	400	13.684 701 0(48)	13.684 742(43)
180	11.015 570 64(39)	11.015 571 6(77)	450	13.707 121 8(38)	13.707 140(37)
200	11.028 931 62(21)	11.028 925 9(46)	500	13.727 027 9(29)	13.727 030(31)
	$N_{\rm tot} = 8000$			$N_{\rm tot} = 32\,000$	
$N_{\rm min}$	$X^{\rm cens}_{\rm PERM}$	$X^{\rm cens}_{\rm jc}$	$N_{\rm min}$	$X^{\rm cens}_{\rm PERM}$	$X^{\rm cens}_{\rm jc}$
2	16.148 415(85)	16.148 470(199)	2	18.922 02(76)	18.922 11(36)
100	16.175 459(77)	16.175 516(184)	100	18.927 96(75)	18.928 04(35)
200	16.203 131(70)	16.203 156(171)	200	18.934 83(73)	18.934 86(34)
300	16.229 231(64)	16.229 317(160)	300	18.941 85(71)	18.941 84(33)
400	16.253 729(58)	16.253 832(150)	400	18.948 89(70)	18.948 84(33)
500	16.276 733(52)	16.276 882(140)	500	18.955 88(68)	18.955 82(32)
600	16.298 361(48)	16.298 492(131)	600	18.962 80(66)	18.962 75(31)
700	16.318 720(43)	16.318 815(123)	700	18.969 64(65)	18.969 60(31)
800	16.337 910(39)	16.337 938(114)	800	18.976 39(63)	18.976 35(30)
900	16.356 014(36)	16.356 073(106)	900	18.983 03(62)	18.982 97(30)
1000	16.373 112(32)	16.373 133(100)	1000	18.989 58(60)	18.989 51(29)
1100	16.389 269(29)	16.389 252(93)	1100	18.996 03(59)	18.995 99(29)
1200	16.404 539(26)	16.404 505(87)	1200	19.002 37(58)	19.002 35(28)
1300	16.418 976(24)	16.418 930(81)	1300	19.008 63(56)	19.008 53(27)
1500	16.445 524(19)	16.445 475(71)	1500	19.020 85(53)	19.020 71(26)
			2000	19.049 78(47)	19.049 73(24)
			2500	19.076 57(42)	19.076 49(22)
			3000	19.101 44(37)	19.101 30(21)

be about ten times slower and therefore less efficient than the join-and-cut algorithm already for $N \approx 500$.

4. Estimate of γ

Let us now discuss the results for the critical exponent γ. We will first analyse the data using the method presented in section 2.2. This will allow us to use at the same time the data produced using the join-and-cut and the PERM. In table 4 we report, for various $N_{\rm min}$, the estimates of $X^{\rm cens}_{\rm MC}(N_{\rm min})$ obtained using the two algorithms. We have checked that the raw data agree within error bars, therefore checking the correctness of our programs and error bars.

In table 5 we report the estimates of the effective exponents $\hat{\gamma}$ defined in section 2.2. A graph of these estimates together with the results of the fits described below is reported in figure 3. It is evident that the corrections to scaling are particularly strong and indeed

Table 5. Estimates of $\gamma(N_{\rm tot})$ for various values of $N_{\rm min}$ and $N_{\rm tot}$.

$N_{\rm min}$	$\gamma(500)$	$N_{\rm min}$	$\gamma(2000)$	$N_{\rm min}$	$\gamma(8000)$	$N_{\rm min}$	$\gamma(32\,000)$
2	1.329 50 (3)	2	1.332 63 (7)	2	1.337 73 (21)	2	1.340 59 (87)
20	1.325 84 (5)	50	1.334 23 (9)	100	1.339 00 (25)	100	1.340 97 (92)
40	1.327 76 (7)	100	1.335 16 (12)	200	1.339 50 (29)	200	1.341 05 (97)
60	1.328 80 (9)	150	1.335 69 (14)	300	1.339 87 (33)	300	1.341 05 (101)
80	1.329 55 (12)	200	1.336 09 (17)	400	1.340 15 (36)	400	1.341 06 (105)
100	1.330 11 (15)	250	1.336 38 (19)	500	1.340 41 (40)	500	1.341 12 (109)
120	1.330 64 (20)	300	1.336 61 (22)	600	1.340 58 (44)	600	1.341 22 (114)
140	1.331 20 (28)	350	1.336 81 (26)	700	1.340 68 (48)	700	1.341 36 (118)
160	1.331 39 (40)	400	1.336 92 (30)	800	1.340 70 (53)	800	1.341 43 (123)
180	1.330 51 (63)	450	1.337 05 (35)	900	1.340 80 (58)	900	1.341 39 (127)
200	1.332 35 (132)	500	1.337 11 (42)	1000	1.340 77 (63)	1000	1.341 43 (131)
				1100	1.340 75 (68)	1100	1.341 62 (136)
				1200	1.340 73 (74)	1200	1.341 72 (139)
				1300	1.340 72 (80)	1300	1.341 47 (143)
				1500	1.340 77 (94)	1500	1.341 40 (153)
						2000	1.342 32 (178)
						2500	1.342 56 (205)
						3000	1.342 50 (236)

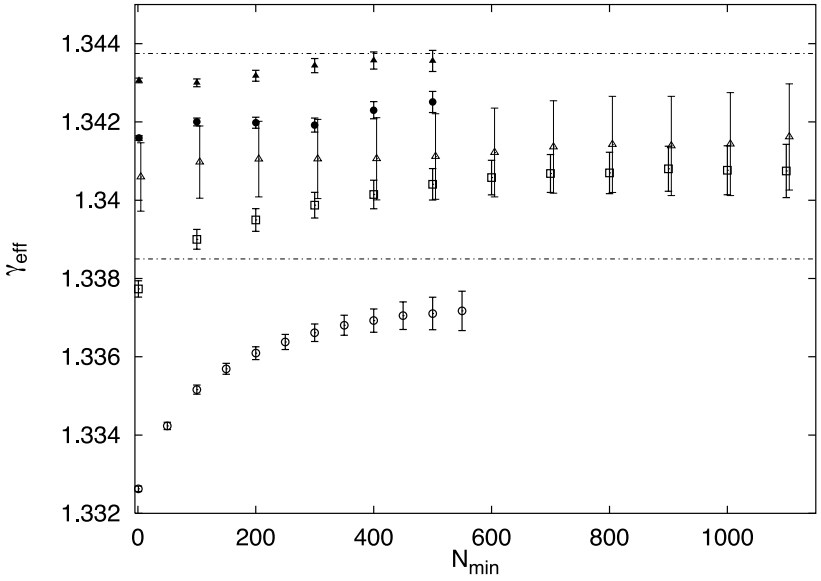

Figure 3. Estimates of the effective exponents $\hat{\gamma}$ for $N_{\rm tot} = 2000$ (empty circle), 8000 (empty square), 32 000 (empty triangle) and estimates obtained with the optimal amplitude method for $\Delta = \frac{11}{16}$ (filled triangle) and $\Delta = 1$ (filled circle). The dotted lines represent $\gamma_{\rm reg} = \frac{43}{32}$ and the estimate of [13], $\gamma = 1.3385$.

$\hat{\gamma}$ clearly increases with $N_{\rm min}$ and $N_{\rm tot}$. Under the only assumption that in the interval $1000 \lesssim N \leqslant 32\,000$ the corrections to scaling have the asymptotic sign, i.e. if we exclude that $\hat{\gamma}$ will decrease for larger values of $N_{\rm min}$ and $N_{\rm tot}$, we obtain (using $N_{\rm tot} = 32\,000$, $N_{\rm min} = 2000$)

$$\gamma \gtrsim 1.3423 \pm 0.0018. \tag{4.1}$$

Table 6. Extrapolated estimates of γ for $\Delta = \frac{11}{16}$ and $\Delta = 1$.

	$\Delta = \frac{11}{16}$		$\Delta = 1$	
$N_{\min}$	γ	$a_{\rm opt}$	γ	$a_{\rm opt}$
2	1.343 06 (6)	0.43 (1)	1.341 59 (6)	0.85 (2)
100	1.343 00 (10)	0.51 (4)	1.342 00 (10)	1.75 (13)
200	1.343 18 (14)	0.54 (4)	1.341 98 (14)	1.93 (24)
300	1.343 44 (18)	0.57 (5)	1.341 92 (18)	1.98 (36)
400	1.343 57 (22)	0.59 (5)	1.342 30 (22)	2.23 (50)
500	1.343 56 (27)	0.59 (6)	1.342 51 (27)	2.40 (59)

This clearly excludes the result in [13]. Under this very weak assumption, we can conclude that there is no evidence from our data of the non-universality predicted in [2]. More conservatively, our data indicate that, if the effect is really there, it is extremely small:

$$\gamma_{\rm reg} - \gamma < 0.0014 \pm 0.0018. \tag{4.2}$$

We can improve this bound making additional assumptions on the corrections to scaling. For each value of $N_{\min}$ we assume that z_N is well approximated by

$$z_N = A\mu^N N^{\gamma-1}(1 + aN^{-\Delta}) \tag{4.3}$$

for $N \geqslant N_{\min}$. We are not able to determine Δ reliably and therefore we have performed fits for various fixed values of Δ. Our analysis works as follows. For each triple $(N_{\min}, N_{\rm tot}, a)$ and for each value of Δ, we can define an effective exponent $\hat{\gamma}(N_{\min}, N_{\rm tot}, a)$ (from now on we suppress the dependence on Δ) using the natural generalization of equations (2.10) and (2.11) with a corresponding error $\Delta\hat{\gamma}(N_{\min}, N_{\rm tot}, a)$. Then for each $N_{\min}$ and Δ we determine the optimal value $a_{\rm opt}$ of a and an estimate of the exponent $\bar{\gamma}(N_{\min})$, by minimizing with respect to a and $\bar{\gamma}(N_{\min})$

$$\chi^2 = \sum_{N_{\rm tot}} \left[\frac{\hat{\gamma}(N_{\min}, N_{\rm tot}, a) - \bar{\gamma}(N_{\min})}{\Delta\hat{\gamma}(N_{\min}, N_{\rm tot}, a)} \right]^2 \tag{4.4}$$

where the sum runs over $N_{\rm tot} = 2000, 8000, 32\,000$. The statistical errors are obtained in a standard fashion.

In table 6 we report the results of the fits for various values of $N_{\min}$ and for two values of Δ, $\Delta = 1$ and $\Delta = \frac{11}{16}$. First let us observe that the estimate of γ decreases with increasing Δ. Therefore a lower bound independent of any assumption on the value of Δ is obtained using the results for $\Delta = 1$. From table 6 we can estimate

$$\gamma \gtrsim 1.3425 \pm 0.0003. \tag{4.5}$$

We can try to do more and see if we can obtain from the data a rough indication of the value of Δ. In principle we can consider the results for various $N_{\min}$ and see for which value of Δ the estimates do not depend on $N_{\min}$. Notice however that the results with different $N_{\min}$ are strongly correlated—they are obtained essentially from the same data—and therefore an observed stability or upward trend should be interpreted with caution since it could simply be an effect of the correlations. If we consider the results of table 6 we see that the analysis with $\Delta = \frac{11}{16}$ is extremely stable: $a_{\rm opt}$ and γ are essentially constant and we can estimate

$$\gamma = 1.3436 \pm 0.0003. \tag{4.6}$$

This result is in perfect agreement with the universal value $\gamma_{\rm reg} = \frac{43}{32}$. On the other hand the results for $\Delta = 1$ show an upward systematic trend. However, the effect is barely statistically

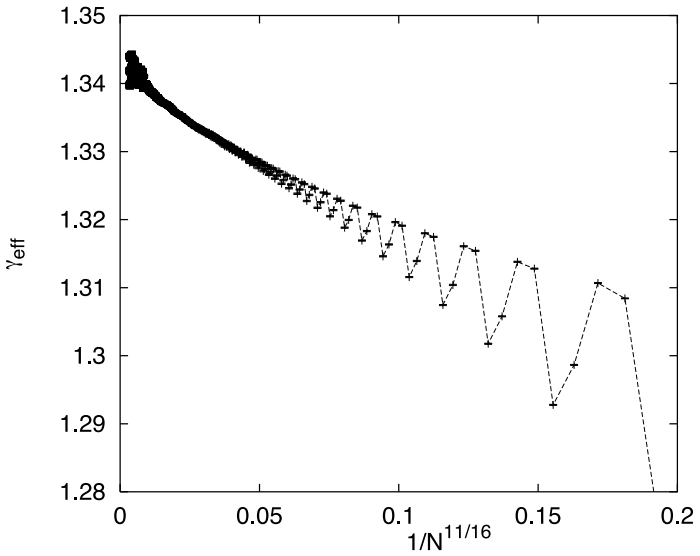

Figure 4. Effective exponents γ_{eff} defined by equation (3.3), plotted against $N^{-11/16}$. The extrapolation to the y-axis gives the estimate of γ.

significant (the results for $N_{\min} = 100$ and $N_{\min} = 500$ differ approximately by 1.5 combined error bars) and one could just think that the systematic increase is simply an effect due to the neglected corrections to scaling and/or a result of the correlations. Without further hypotheses, we cannot confidently go beyond the lower bound (4.5). However, if we assume $\gamma = \gamma_{\text{reg}}$, then $\Delta = 1$ becomes less plausible and we can say that our data favour the presence of a non-analytic exponent. We cannot give a reliable estimate of Δ, but Saleur's value $\frac{11}{16}$ fits our data very well.

This conclusion is fully supported by the alternative analysis of the PERM data using equation (3.3). In figure 4 we plot γ_{eff}, obtained with $a = \frac{1}{2}$, $b = 4$, against $N^{-11/16}$. If Δ has Saleur's value, and if there are no other corrections to scaling, we should expect a straight line intersecting the y-axis at γ. The most dramatic deviations from a straight line are strong period-four oscillations, also observed in [13]. Similar period-four oscillations are observed for SAWs on the square lattice [9]. They correspond to a singularity of the grand canonical partition sum at $x = -1/\mu$ [39]. In figure 4 one may also observe a slight curvature which might suggest that $\Delta < \frac{11}{16}$. However, a more careful analysis, also using different pairs (a, b) and additional correction-to-scaling terms, suggests that this effect is not significant. In contrast, the fact that $\Delta < 1$ seems significant. Accepting for Δ a value between $\frac{1}{2}$ and 0.7, we find again perfect agreement with $\gamma_{\text{reg}} = \frac{43}{32}$, while the estimate in [13] seems ruled out.

Acknowledgments

We thank John Cardy, Tony Guttmann, Flavio Seno and Antonio Trovato for very helpful discussions.

References

[1] Miller J 1991 *J. Stat. Phys.* **63** 89

[2] Cardy J L 1994 *Nucl. Phys.* B **419** 411
[3] Bennet-Wood D, Cardy J L, Flesia S, Guttmann A J and Owczarek A L 1995 *J. Phys. A: Math. Gen.* **28** 5143
[4] Barkema G T and Flesia S 1996 *J. Stat. Phys.* **85** 363
[5] Trovato A and Seno F 1997 *Phys. Rev.* E **56** 131
[6] Prellberg T and Drossel B 1998 *Phys. Rev.* E **57** 2045
[7] Barkema G T, Bastolla U and Grassberger P 1998 *J. Stat. Phys.* **90** 1311
[8] Nienhuis B 1982 *Phys. Rev. Lett.* **49** 1062
Nienhuis B 1984 *J. Stat. Phys.* **34** 731
[9] Conway A, Enting I G and Guttmann A J 1993 *J. Phys. A: Math. Gen.* **26** 1519
[10] Koo W M 1995 *J. Stat. Phys.* **81** 561
[11] Flesia S 1995 *Europhys. Lett.* **32** 149
[12] Frauenkron H, Causo M S and Grassberger P 1999 *Phys. Rev.* E **59** R16
(Frauenkron H, Causo M S and Grassberger P 1998 2-d self-avoiding walks on a cylinder *Preprint* cond-mat/9808075)
[13] Bennet-Wood D, Cardy J L, Enting I G, Guttmann A J and Owczarek A L 1998 *Nucl. Phys.* B **528** 533
[14] Caracciolo S, Pelissetto A and Sokal A D 1992 *J. Stat. Phys.* **67** 65
[15] Grassberger P 1997 *Phys. Rev.* E **56** 3682
[16] Guttmann A J and Enting I G 1988 *J. Phys. A: Math. Gen.* **21** L165
[17] Conway A R and Guttmann A J 1996 *Phys. Rev. Lett.* **77** 5284
[18] Bennett-Wood D, Caracciolo S, Grassberger P, Guttmann A J, Li B, Pelissetto A and Sokal A D in preparation
[19] Saleur H 1987 *J. Phys. A: Math. Gen.* **20** 455
[20] Conway A R and Guttmann A J 1993 *J. Phys. A: Math. Gen.* **26** 1535
[21] Guim I, Blöte H W J and Burkhardt T W 1997 *J. Phys. A: Math. Gen.* **30** 413
[22] Guttmann A J Private communication
[23] Lal M 1969 *Mol. Phys.* **17** 57
[24] MacDonald B, Jan N, Hunter D L and Steinitz M O 1985 *J. Phys. A: Math. Gen.* **18** 2627
[25] Madras N and Sokal A D 1988 *J. Stat. Phys.* **50** 109
[26] Caracciolo S, Parisi G and Pelissetto A 1994 *J. Stat. Phys.* **77** 519
[27] Belohorec P and Nickel B G 1997 Accurate universal and two-parameter model results from a Monte-Carlo renormalization group study *Guelph University Report (September 1997)*
[28] Zifferer G 1993 *Makromol. Chemie Theory Simul.* **2** 653
[29] Tesi M C, Janse van Rensburg E J, Orlandini E and Whittington S G 1996 *J. Stat. Phys.* **82** 155
[30] Madras N, Orlitsky A and Shepp L A 1990 *J. Stat. Phys.* **58** 159
[31] Silvey S D 1975 *Statistical Inference* (London: Chapman and Hall)
[32] Li B, Madras N and Sokal A D 1995 *J. Stat. Phys.* **80** 661
[33] Madras N and Slade G 1993 *The Self-Avoiding Walk* (Berlin: Birkhäuser)
[34] Meirovitch H and Lim H A 1990 *J. Chem. Phys.* **92** 5144
[35] Grassberger P to be published
[36] Redner S and Reynolds P J 1981 *J. Phys. A: Math. Gen.* **14** 2679
[37] Berretti A and Sokal A D 1985 *J. Stat. Phys.* **40** 483
[38] Grassberger P, Sutter P and Schäfer L 1997 *J. Phys. A: Math. Gen.* **30** 7039
[39] Guttmann A J and Whittington S G 1978 *J. Phys. A: Math. Gen.* **11** 721