Configurational-Bias Monte Carlo

Thijs J.H. Vlugt

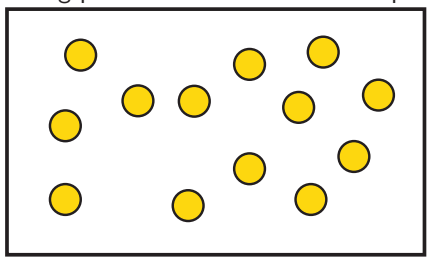
Engineering Thermodynamics, Process & Energy Department
Faculty of Mechanical, Maritime and Materials Engineering (3mE)
Delft University of Technology
Delft, The Netherlands

t.j.h.vlugt@tudelft.nl

January 9, 2009

Random Sampling versus Metropolis Sampling (1)

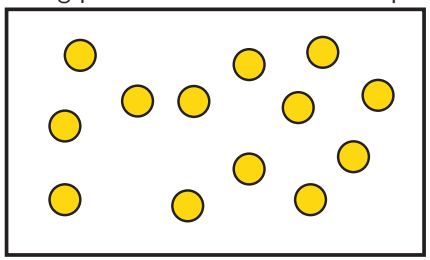
N interacting particles in volume V at temperature T



- vector representing positions of all particles in the system: $\mathbf{r}^N$
- total energy: $U(\mathbf{r}^N)$
- statistical weight of configuration $\mathbf{r}^N$ is $\exp[-\beta U(\mathbf{r}^N)]$ with $\beta = 1/(k_B T)$

Random Sampling versus Metropolis Sampling (2)

N interacting particles in volume V at temperature T



$$
\begin{aligned}
Q(N,V,T) &= \frac{1}{\Lambda^{3N} N!} \int d\mathbf{r}^N \exp\left[-\beta U(\mathbf{r}^N)\right] \\
F(N,V,T) &= -k_B T \ln Q(N,V,T) \\
U(\mathbf{r}^N) &= \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} u(r_{ij}) = \sum_{i<j} u(r_{ij})
\end{aligned}
$$

Random Sampling versus Metropolis Sampling (3)

Computing the ensemble average $\langle \cdots \rangle$ of a certain quantity $A(\mathbf{r}^N)$

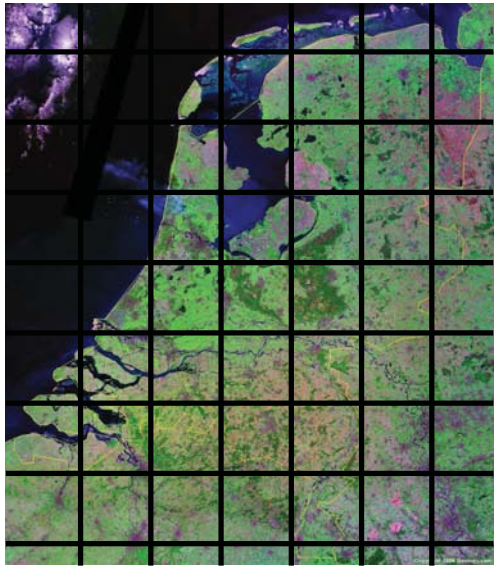
- Random Sampling of $\mathbf{r}^N$:

$$\langle A \rangle = \lim_{n\to\infty} \frac{\sum_{i=1}^{n} A(\mathbf{r}_i^N) \exp\left[-\beta U(\mathbf{r}_i^N)\right]}{\sum_{i=1}^{n} \exp\left[-\beta U(\mathbf{r}_i^N)\right]}$$

- Metropolis sampling; generate n configurations $\mathbf{r}^N$ with probability proportional to $\exp\left[-\beta U(\mathbf{r}_i^N)\right]$, therefore:

$$\langle A \rangle = \lim_{n\to\infty} \frac{\sum_{i=1}^{n} A(\mathbf{r}_i^N)}{n}$$

Simulation Technique (3)



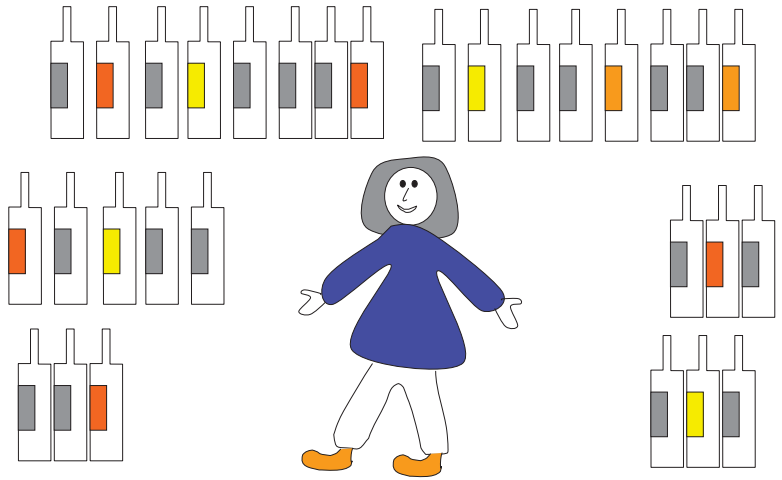
Simulation Technique (1)

What is the ratio of red wine/white wine in the Netherlands?

Simulation Technique (4)

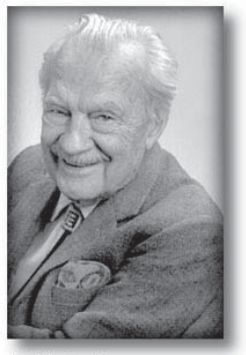
Simulation Technique (2)

Bottoms up



Metropolis Monte Carlo (1)

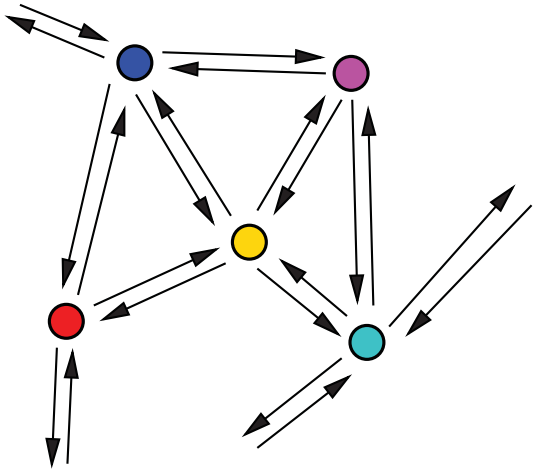
How to generate configurations $\mathbf{r}_i$ with a probability proportional to $\mathcal{N}(\mathbf{r}_i) = \exp[-\beta U(\mathbf{r}_i)]$???


Nick Metropolis

N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth. A.H. Teller and E. Teller, "Equation of State Calculations by Fast Computing Machines," J. Chem. Phys. 21 (1953) 1087-1092.

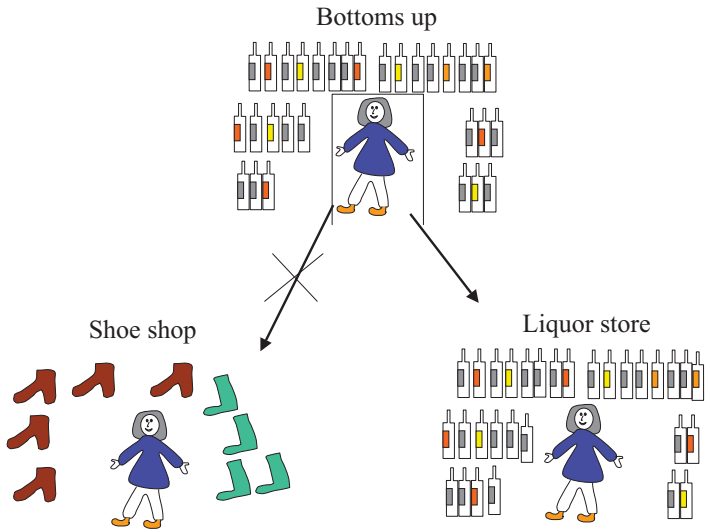
Metropolis Monte Carlo (2)

Whatever our rule is to move from one state to the next, the equilibrium distribution should not be destroyed



Simulation Technique (5)

Simulation Technique (6)



Detailed Balance (2)

$$\mathcal{N}(o)\alpha(o \to n)\mathrm{acc}(o \to n) = \mathcal{N}(n)\alpha(n \to o)\mathrm{acc}(n \to o)$$

- $\alpha(x \to y)$; probability to select move from x to y
- $\mathrm{acc}(x \to y)$; probability to accept move from x to y
- often (but not always); $\alpha(o \to n) = \alpha(n \to o)$

Therefore:

$$\frac{\mathrm{acc}(o \to n)}{\mathrm{acc}(n \to o)} = \frac{\alpha(n \to o)\exp[-\beta U(n)]}{\alpha(o \to n)\exp[-\beta U(o)]} = \frac{\alpha(n \to o)}{\alpha(o \to n)}\exp[-\beta \Delta U]$$

Metropolis Acceptance Rule

General:

$$\frac{\mathrm{acc}(o \to n)}{\mathrm{acc}(n \to o)} = X$$

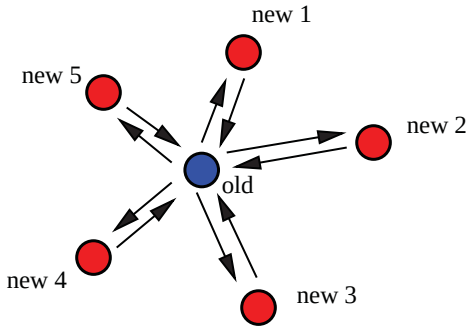
Choice made by Metropolis (note: infinite number of other possibilities)

$$\mathrm{acc}(o \to n) = \min(1, X)$$

Note than $\min(a, b) = a$ if $a < b$ and b otherwise

- always accept when $X \geq 1$
- when $X < 1$, generate uniformly distributed random number from $< 0, 1 >$ and accept or reject according to $\mathrm{acc}(o \to n)$

Move from the old state (o) to a new state (n) and back



$$\begin{aligned} \text{leaving state o} &= \text{entering state o} \\ \mathcal{N}(o)\sum_n [\alpha(o \to n)\mathrm{acc}(o \to n)] &= \sum_n [\mathcal{N}(n)\alpha(n \to o)\mathrm{acc}(n \to o)] \end{aligned}$$

$\mathcal{N}(i)$: probability to be in state i (here: proportional to $\exp[-\beta E_i]$)
$\alpha(x \to y)$: probability to attempt move from state x to state y
$\mathrm{acc}(x \to y)$: probability to accept move from state x to state y

Detailed Balance (1)

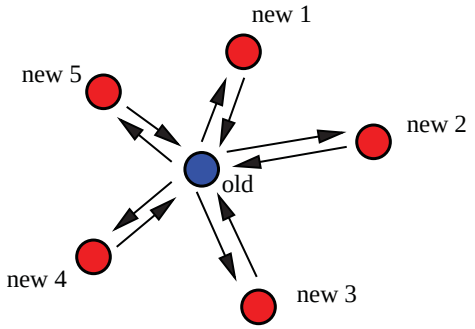
Requirement (balance):

$$\mathcal{N}(o)\sum_n [\alpha(o \to n)\mathrm{acc}(o \to n)] = \sum_n [\mathcal{N}(n)\alpha(n \to o)\mathrm{acc}(n \to o)]$$

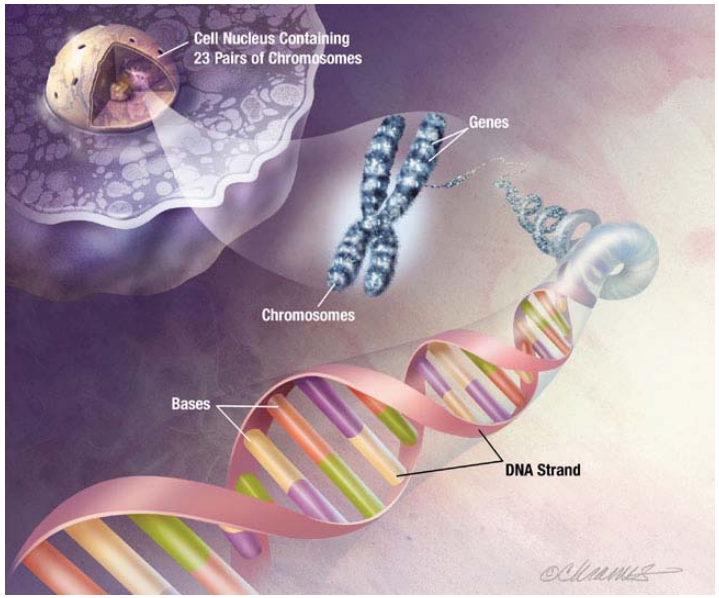
Detailed balance: much stronger condition

$$\mathcal{N}(o)\alpha(o \to n)\mathrm{acc}(o \to n) = \mathcal{N}(n)\alpha(n \to o)\mathrm{acc}(n \to o)$$

for every pair o,n

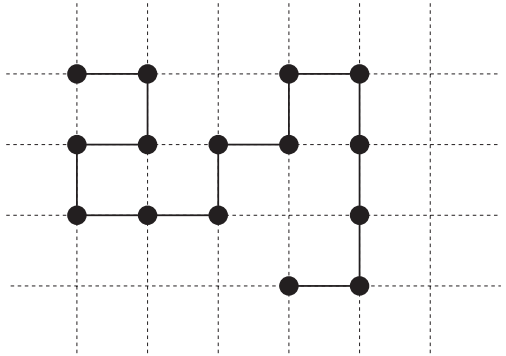


Chain Molecules



Monte Carlo Casino

Self-Avoiding Walk on a Cubic Lattice



- 3D lattice; 6 lattice directions
- only 1 monomer per lattice site (otherwise $U = \infty$)
- interactions when $|r_{ij}| = 1$ and $|i - j| > 1$

Smart Monte Carlo: $\alpha(o \rightarrow n) \neq \alpha(n \rightarrow o)$

Not a random displacement Δr uniformly from $[-\delta, \delta]$, but instead

$$\Delta r = r(\text{new}) - r(\text{old}) = A \times F + \delta r$$

F : force on particle
A : constant
δr : taken from Gaussian distribution with width $2A$
so $P(\delta r) \propto \exp[-(\delta r^2)/4A]$

$$P(r_{\text{new}}) \propto \exp\left[-\frac{(r_{\text{new}} - (r_{\text{old}} + A \times F(o)))^2}{4A}\right]$$

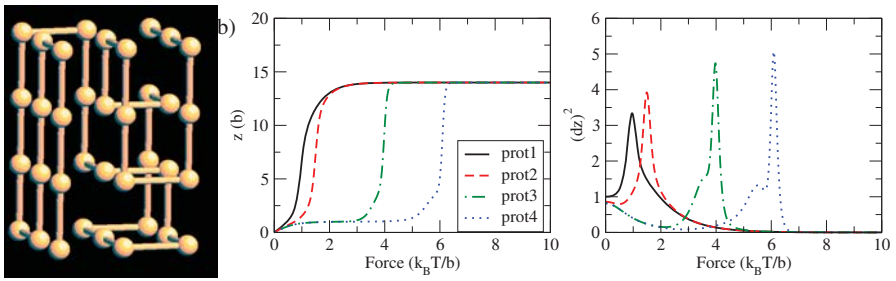
$$\frac{\alpha(o \rightarrow n)}{\alpha(n \rightarrow o)} = \frac{\exp\left[-\frac{(\Delta r - A \times F(o))^2}{4A}\right]}{\exp\left[-\frac{(\Delta r + A \times F(n))^2}{4A}\right]}$$

Simple Model for Protein Folding

20 by 20 interaction matrix Δ_{ij}

YPDLTKWHAMEAGKIRFSVPDACLNGEGIRQVTLSN

(E. Jarkova, T.J.H. Vlugt, N.K. Lee, J. Chem. Phys., 2005, 122, 114904)



Number of Configurations without Overlap

Random Chains:

$$\langle R \rangle = \lim_{n\to\infty} \frac{\sum_{i=1}^{n} R_i \exp[-\beta U_i]}{\sum_{i=1}^{n} \exp[-\beta U_i]}$$

Fraction of chains without overlap decreases exponentially as a function of chainlength (N)

N	total ($=6^{N-1}$)	without overlap	fraction no overlap
2	6	6	1.0
6	7776	3534	0.454
8	279936	81390	0.290
10	10077696	1853886	0.183
12	362797056	41934150	0.115
13	2176782336	198842742	0.091
14	13060694016	943974510	0.072
15	78364164096	4468911678	0.057
16	470184984576	21175146054	0.045
50	$\cdots$	$\cdots$	1.3×10^{-5}

Rosenbluth Sampling (1)

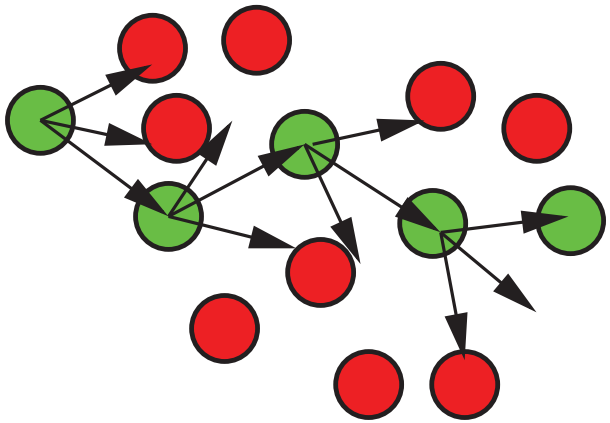
1. Place first monomer at a random position
2. For the next monomer (i), generate k trial directions ($j = 1 \cdots k$) each with energy u_{ij}
3. Select trial direction $j^\star$ with a probability

$$P_{j^\star} = \frac{\exp[-\beta u_{ij^\star}]}{\sum_{j=1}^{k} \exp[-\beta u_{ij}]}$$

4. Continue with step 2 until the complete chain is grown (N monomers)

Rosenbluth Sampling (2)

$$P_{j^\star} = \frac{\exp[-\beta u_{ij^\star}]}{\sum_{j=1}^{k} \exp[-\beta u_{ij}]}$$



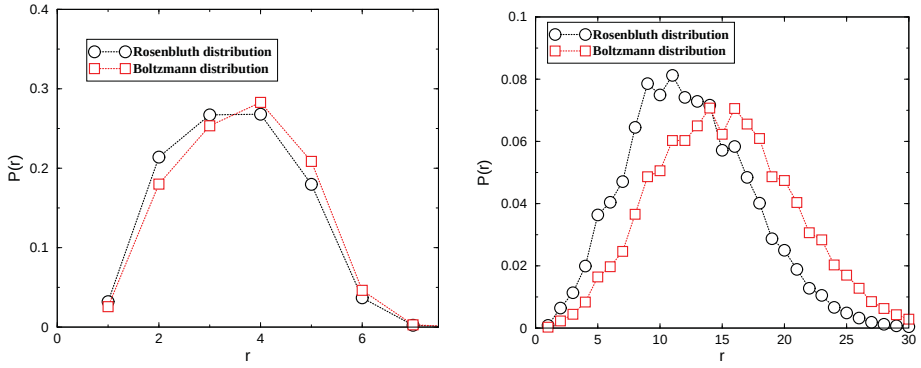
Intermezzo: Ensemble Averages at Different Temperatures

$$
\begin{aligned}
\langle U \rangle_\beta &= \frac{\int d\mathbf{r}^N U(\mathbf{r}^N) \exp[-\beta U(\mathbf{r}^N)]}{\int d\mathbf{r}^N \exp[-\beta U(\mathbf{r}^N)]} \\
&= \frac{\int d\mathbf{r}^N U(\mathbf{r}^N) \exp[-\beta^\star U(\mathbf{r}^N)] \exp[(\beta^\star - \beta) \times U(\mathbf{r}^N)]}{\int d\mathbf{r}^N \exp[-\beta^\star U(\mathbf{r}^N)] \exp[(\beta^\star - \beta) \times U(\mathbf{r}^N)]} \\
&= \frac{\left\langle U(\mathbf{r}^N) \exp[(\beta^\star - \beta) \times U(\mathbf{r}^N)] \right\rangle_{\beta^\star}}{\left\langle \exp[(\beta^\star - \beta) \times U(\mathbf{r}^N)] \right\rangle_{\beta^\star}} \\
&= \frac{\left\langle U(\mathbf{r}^N) \exp[\Delta\beta \times U(\mathbf{r}^N)] \right\rangle_{\beta^\star}}{\left\langle \exp[\Delta\beta \times U(\mathbf{r}^N)] \right\rangle_{\beta^\star}}
\end{aligned}
$$

Rosenbluth Distribution Differs from Boltzmann Distribution

end to end distance r

$N = 10$ $N = 100$



Rosenbluth Sampling (3)

Probability to insert monomer i (at trial direction $j^\star$)

$$P_{j^\star} = \frac{\exp[-\beta u_{ij^\star}]}{\sum_{j=1}^{k} \exp[-\beta u_{ij}]}$$

Probability to grow the chain (N monomers, k trial directions)

$$P_{\text{chain}} = \frac{\prod_{i=1}^{N} \exp[-\beta u_{ij^\star(i)}]}{\prod_{i=1}^{N} \sum_{j=1}^{k} \exp[-\beta u_{ij}]} = \frac{\exp[-\beta U_{\text{chain}}]}{W_{\text{chain}}}$$

Rosenbluth Sampling (4)

Probability to grow the chain (N monomers, k trial directions)

$$P_{\text{chain}} = \frac{\prod_{i=1}^{N} \exp[-\beta u_{ij^\star(i)}]}{\prod_{i=1}^{N} \sum_{j=1}^{k} \exp[-\beta u_{ij}]} = \frac{\exp[-\beta U_{\text{chain}}]}{W_{\text{chain}}}$$

Therefore, weightfactor for each chain i is the Rosenbluth factor W_i:

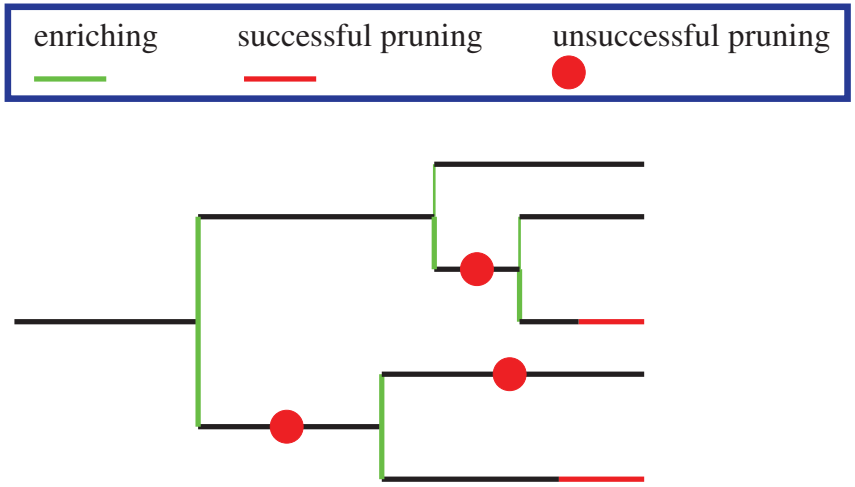
$$\langle R \rangle_{\text{Boltzmann}} = \lim_{n\to\infty} \frac{\sum_{i=1}^{n} W_i \times R_i}{\sum_{i=1}^{n} W_i}$$

The unweighted distribution is called the Rosenbluth distribution:

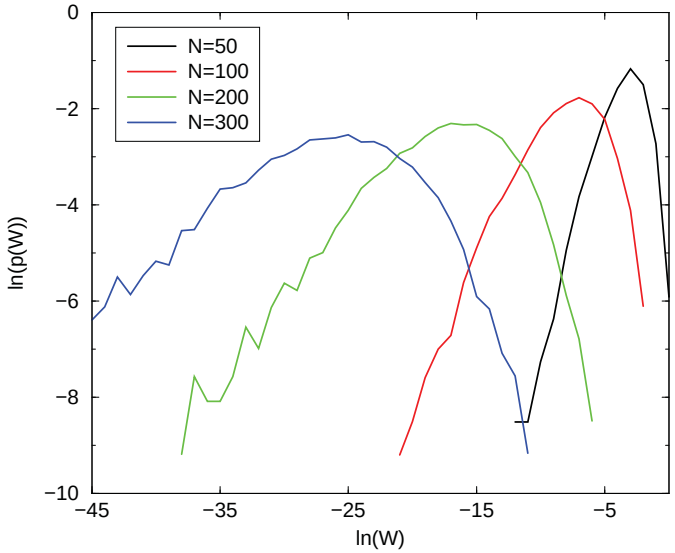
$$\langle R \rangle_{\text{Rosenbluth}} = \lim_{n\to\infty} \frac{\sum_{i=1}^{n} R_i}{n}$$

Of course: $\langle R \rangle_{\text{Rosenbluth}} \neq \langle R \rangle_{\text{Boltzmann}}$

Pruned-Enriched Rosenbluth Method (2)

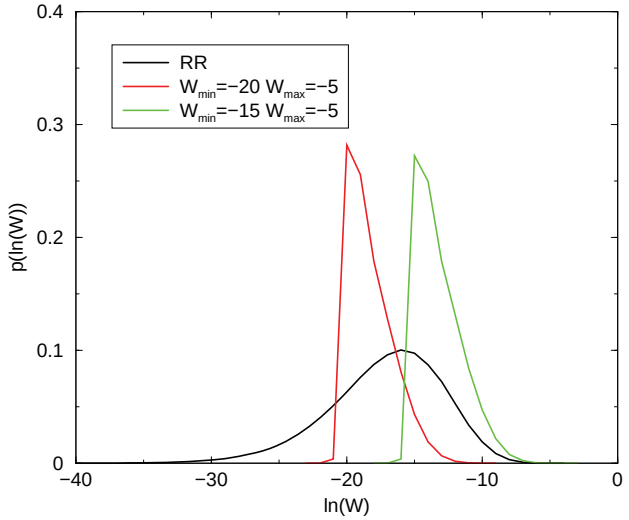


Distribution of Rosenbluth Weights



Pruned-Enriched Rosenbluth Method (3)

Example: $N = 200$, $k = 2$, $\beta\mu_{\rm ex} = -\ln\langle W\rangle = -12.14$



Pruned-Enriched Rosenbluth Method (1)

Grassberger (1997); grow chains using Rosenbluth Method:

$$W = \sum_{j=1}^{6} \frac{\exp\left[-\beta u_{2j}\right]}{6} \times \prod_{i=3}^{N} \sum_{j=1}^{5} \frac{\exp\left[-\beta u_{ij}\right]}{5} = \prod_{i=3}^{N} \sum_{j=1}^{5} \frac{\exp\left[-\beta u_{ij}\right]}{5}$$

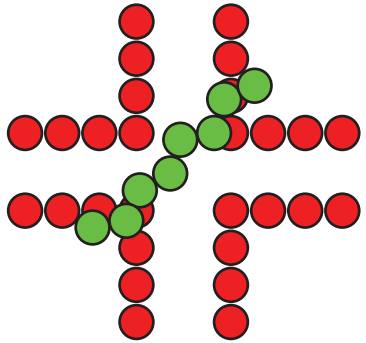
Two additional elements:

- Enriching.
  If $W > W_{\rm max}$ during the construction of the chain, k copies of the chain are generated, each with a weight of W/k. This is a deterministic process. The growth of those chains is continued.

- Pruning.
  If $W < W_{\rm min}$ during the construction of the chain, with a probability of $1/2$ the chain is pruned resulting in $W = 0$. If the chain survives, the Rosenbluth weight is multiplied by 2 and the growth of the chain is continued.

Random Insertion of Chains is Useless !

Chain Length	Probability without overlaps
1	10^{-2}
2	10^{-4}
3	10^{-6}
...	...
8	10^{-16}



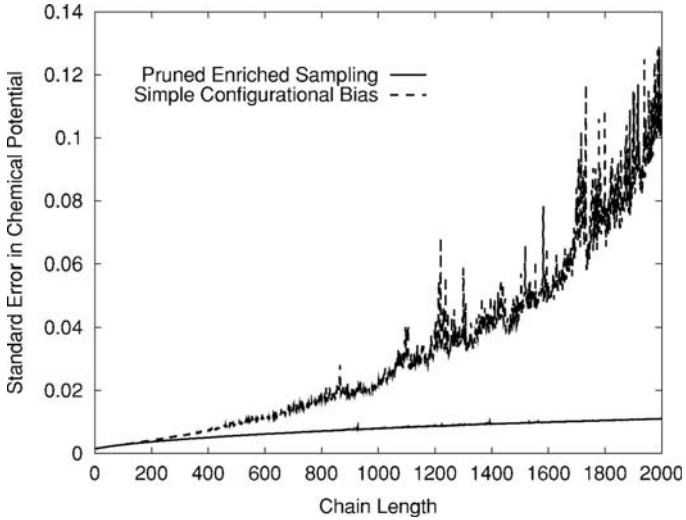
Configurational-Bias Monte Carlo

- Generate configurations using the Rosenbluth scheme
- Accept/Reject these configurations in such a way that detailed balance is obeyed
- Split potential energy into "**bonded**" (bond-stretching, bending, torsion) and "**non-bonded**" (i.e. Lennard-Jones or Coulombic) interactions
- Generate (k) trial positions according to **bonded** interactions (unbranched chain: l, θ, ϕ are independent)

$$
\begin{aligned}
U_{\text{bonded}} &= U_{\text{stretch}}(l) + U_{\text{bend}}(\theta) + U_{\text{tors}}(\phi) \\
P(l) &\propto dl\, l^2 \exp[-\beta u(l)] \\
P(\theta) &\propto d\theta\, \sin(\theta) \exp[-\beta u(\theta)] \\
P(\phi) &\propto d\phi\, \exp[-\beta u(\phi)]
\end{aligned}
$$

Pruned-Enriched Rosenbluth Method (4)

$\beta\mu_{\text{ex}} = -\ln\langle W\rangle$, Ann. Rev. of Phys. Chem. 1999, 50, 377-411



Static versus Dynamic

- **Static**
  - create chain conformations, and use correct weight factor (random insertion, Rosenbluth scheme, PERM)
  - **single chain only; no Markov chain**
- **Dynamic**
  - create Markov chain, accept new configuration, acceptance rule should obey detailed balance
  - **multi-chain system, usable for all ensembles (i.e. Gibbs, μVT)**
  - Configurational-Bias Monte Carlo (CBMC), Recoil Growth (RG), Dynamic PERM (DPERM)

Generate a Random Number from a Distribution (2)

$$F(y) = \int_0^y f(y')dy' \qquad f(x) > p(x)$$

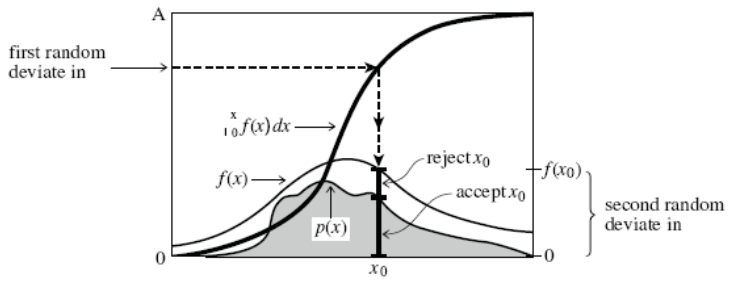


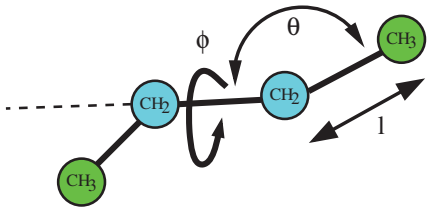
Figure 7.3.1. Rejection method for generating a random deviate x from a known probability distribution $p(x)$ that is everywhere less than some other function $f(x)$. The transformation method is first used to generate a random deviate x of the distribution f (compare Figure 7.2.1). A second uniform deviate is used to decide whether to accept or reject that x. If it is rejected, a new deviate of f is found; and so on. The ratio of accepted to rejected points is the ratio of the area under p to the area between p and f.

Generate a Random Number from a Distribution (3)

```
subroutine bend-tors                          generate appropriate θ and φ

lready=.false
do while (.not.lready)
  call ransphere(dx,dy,dz)                    random vector on unit sphere
  x = xold + dx                               monomer position
  y = yold + dy
  z = zold + dz
  call bend(ubend,x,y,z)                      bending energy
  call tors(utors,x,y,z)                      torsion energy
  if(ranf().lt.exp(-beta*(ubend+utors)))      accept or reject
 +  lready=.true
enddo
```


Generate Trial Configurations: Linear Alkane



$$\begin{aligned}
u(l) &= \frac{k_l}{2}(l-l_0)^2 \\
u(\theta) &= \frac{k_\theta}{2}(\theta-\theta_0)^2 \\
u(\phi) &= \sum_{i=0}^{5} c_i \cos^i(\phi) \\
P(l) &\propto dl\, l^2 \exp[-\beta u(l)] \\
P(\theta) &\propto d\theta\, \sin(\theta) \exp[-\beta u(\theta)] \\
P(\phi) &\propto d\phi\, \exp[-\beta u(\phi)]
\end{aligned}$$

Generate a Random Number from a Distribution (1)

$$F(y) = \int_0^y p(y')dy'$$

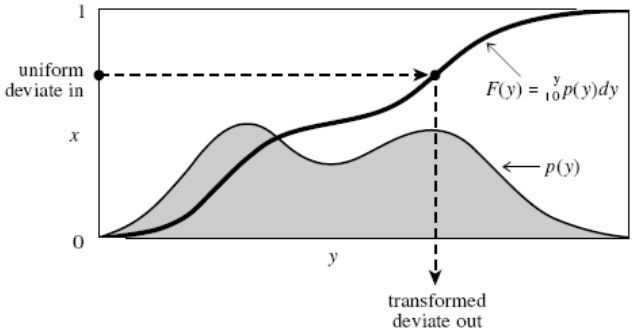


Figure 7.2.1. Transformation method for generating a random deviate y from a known probability distribution $p(y)$. The indefinite integral of $p(y)$ must be known and invertible. A uniform deviate x is chosen between 0 and 1. Its corresponding y on the definite-integral curve is the desired deviate.

Super-Detailed Balance (2)

Notation: $K(o \to n) = \mathcal{N}(o)\alpha(o \to n)\mathrm{acc}(o \to n)$

$$\begin{aligned} K(o \to n) &= \sum_{\mathbf{b}_n \mathbf{b}_o} K(o \to n|\mathbf{b}_n \mathbf{b}_o) \\ K(n \to o) &= \sum_{\mathbf{b}_n \mathbf{b}_o} K(n \to o|\mathbf{b}_n \mathbf{b}_o) \end{aligned}$$

Detailed balance requires

$$K(o \to n) = K(n \to o)$$

Possible solution (Super-detailed balance):

$$K(o \to n|\mathbf{b}_n \mathbf{b}_o) = K(n \to o|\mathbf{b}_n \mathbf{b}_o)$$

for each $\mathbf{b}_n \mathbf{b}_o$.

Super-Detailed Balance (3)

Detailed balance for each set $\mathbf{b}_n \mathbf{b}_o$:

$$\begin{aligned} K(o \to n|\mathbf{b}_n \mathbf{b}_o) &= N(o) \times \alpha(o \to n|\mathbf{b}_n \mathbf{b}_o) \times \mathrm{acc}(o \to n|\mathbf{b}_n \mathbf{b}_o) \\ &= \exp[-\beta U(o)] \times C \exp[-\beta u_{\mathrm{bonded}}(n)] \times \\ &\quad \frac{\exp[-\beta u_{\mathrm{non-b}}(n)]}{W(n)} \times P(\mathbf{b}_n \mathbf{b}_o) \times \mathrm{acc}(o \to n|\mathbf{b}_n \mathbf{b}_o) \\ K(n \to o|\mathbf{b}_n \mathbf{b}_o) &= \exp[-\beta U(n)] \times C \exp[-\beta u_{\mathrm{bonded}}(o)] \times \\ &\quad \frac{\exp[-\beta u_{\mathrm{non-b}}(o)]}{W(o)} \times P(\mathbf{b}_n \mathbf{b}_o) \times \mathrm{acc}(n \to o|\mathbf{b}_n \mathbf{b}_o) \end{aligned}$$

As

$$U = U_{\mathrm{non-b}} + U_{\mathrm{bonded}}$$

therefore

$$\frac{\mathrm{acc}(o \to n)}{\mathrm{acc}(n \to o)} = \frac{W(n)}{W(o)}$$

CBMC Algorithm

- Generate a trial configuration using the Rosenbuth scheme. k trial segments $\{\mathbf{b}\}_k = \{\mathbf{b}_1 \cdots \mathbf{b}_k\}$, each trial segment is generated according to

$$P(\mathbf{b}) \propto \exp[-\beta u_{\mathrm{bonded}}(\mathbf{b})]$$

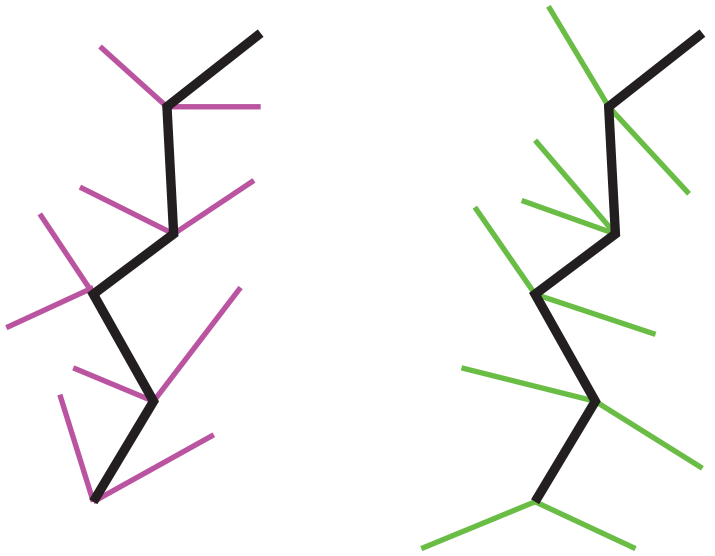
- Compute non-bonded energy, select configuration i with a probability

$$P(\mathbf{b}_i) = \frac{\exp[-\beta u_{\mathrm{non-b}}(\mathbf{b}_i)]}{\sum_{j=1}^{k} \exp[-\beta u_{\mathrm{non-b}}(\mathbf{b}_j)]} = \frac{\exp[-\beta u_{\mathrm{non-b}}(\mathbf{b}_i)]}{w_l}$$

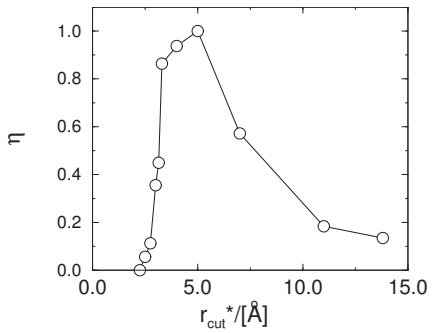
- Continue until chain is grown, $W(n) = \prod_{l=1}^{n} w_l$
- Similar procedure for old configuration, generate $k-1$ trial positions (trial position 1 is the old configuration itself), leading to $W(o)$
- Accept/reject according to

$$\mathrm{acc}(o \to n) = \min(1, W(n)/W(o))$$

Super-Detailed Balance (1)



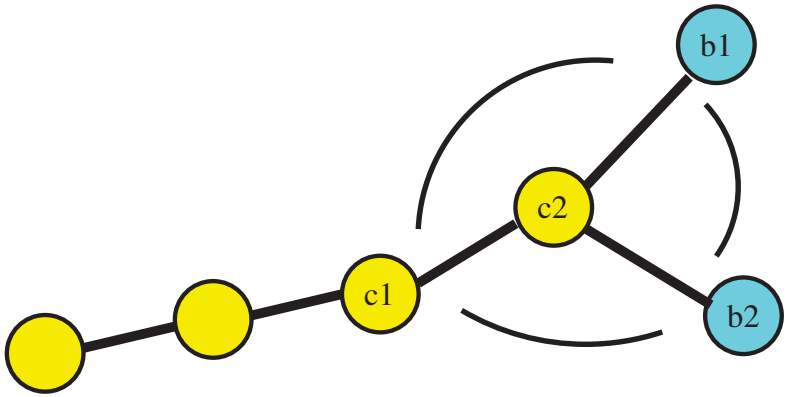
Trick: Dual-Cutoff CBMC



- Grow chain with approximate potential; $W^\star$
- Correct for difference later
  (δu, difference real and approximate potential for *selected* configuration)

$$\mathrm{acc}(o \rightarrow n) = \min\left(1, \frac{W^\star(n)}{W^\star(o)} \times \exp[-\beta[\delta u(n) - \delta u(o)]]\right)$$

Branched Molecules (1)



$$P(\mathbf{b}_1, \mathbf{b}_2) \propto \exp[-\beta[u_{\mathrm{bend}}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{b}_1)] + u_{\mathrm{bend}}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{b}_2)] + u_{\mathrm{bend}}(\mathbf{b}_1, \mathbf{c}_2, \mathbf{b}_2)]]$$

Application 1: Adsorption in MFI-type zeolite (1)

Zeolites:

- microporous channel structure
- crystalline, SiO_2 building blocks
- substitution of Si^{4+} by Al^{3+} and a cation (Na^+ or H^+)
- typical poresize: $4 - 12\mathring{A}$
- synthetic and natural; >170 framework types

Branched Molecules (2)

Use CBMC to generate internal configurations:

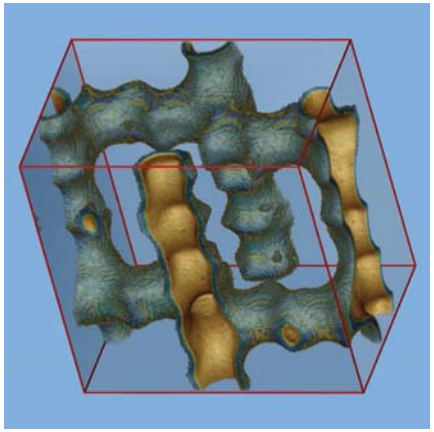
- Generate n_t *random* trial positions and select one (i) with a probability

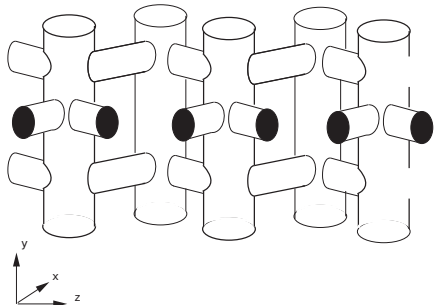
$$P^{\mathrm{int}}(i) = \frac{\exp[-\beta U_{\mathrm{bonded}}(i)]}{\sum_{j=1}^{n_t} \exp[-\beta U_{\mathrm{bonded}}(j)]} = \frac{\exp[-\beta U_{\mathrm{bonded}}(i)]}{W^{\mathrm{int}}(n)}$$

- Repeat until k trial orientations are found; these are fed into CBMC leading to $W(n)$
- Repeat procedure for old configuration, leading to $W^{\mathrm{int}}(o)$ and $W(o)$.
- Accept or reject according to

$$\mathrm{acc}(o \rightarrow n) = \min\left(1, \frac{W(n) \times W^{\mathrm{int}}(n)}{W(o) \times W^{\mathrm{int}}(o)}\right)$$

Application 1: Adsorption in MFI-type zeolite (2)

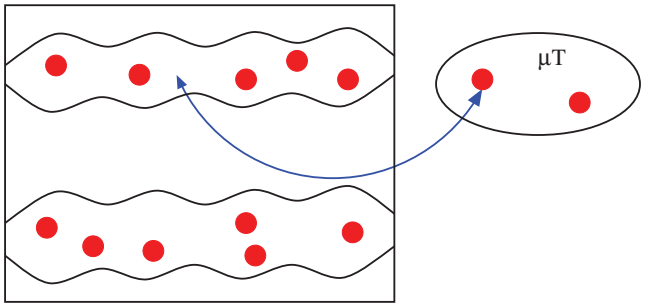




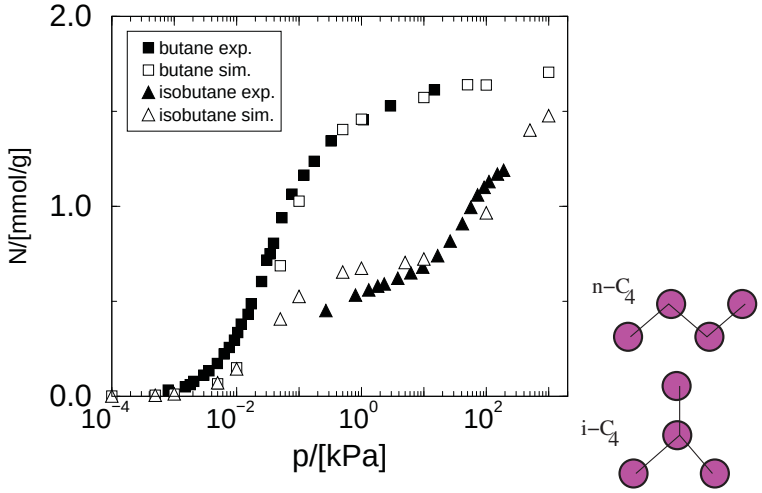
Application 1: Adsorption in MFI-type zeolite (3)

Grand-canonical ensemble

- system is coupled to particle reservoir at chemical potential μ and temperature T, statistical weight $\propto \exp[\beta\mu N - \beta U(\mathbf{r}^N)]/N!$
- additional trial moves to exchange particles between the zeolite and the reservoir
- equilibrium: $\mu_{\rm gas} = \mu_{\rm zeolite}$; measure average $\langle N \rangle$ for given μ and β

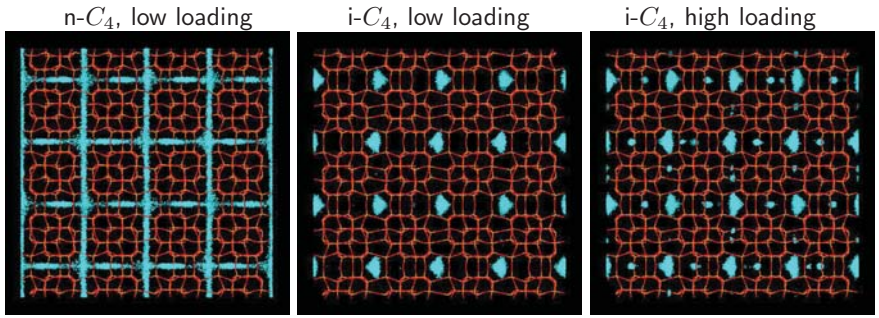


Application 1: Adsorption in MFI-type zeolite (4)



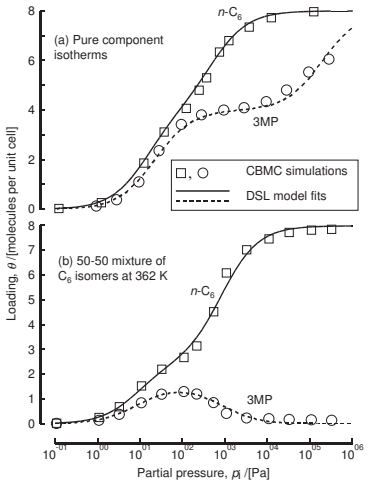
Vlugt et al, J. Am. Chem. Soc., 1998, 120, 5599

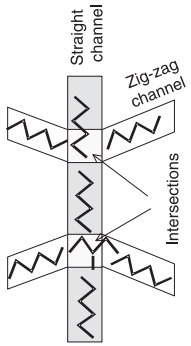
Application 1: Adsorption in MFI-type zeolite (5)



Vlugt et al, J. Am. Chem. Soc., 1998, 120, 5599

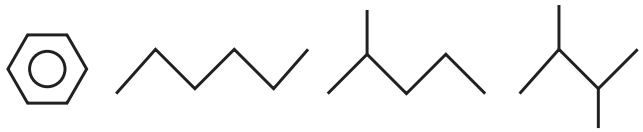
Application 1: Adsorption in MFI-type zeolite (8)





Application 1: Adsorption in MFI-type zeolite (6)

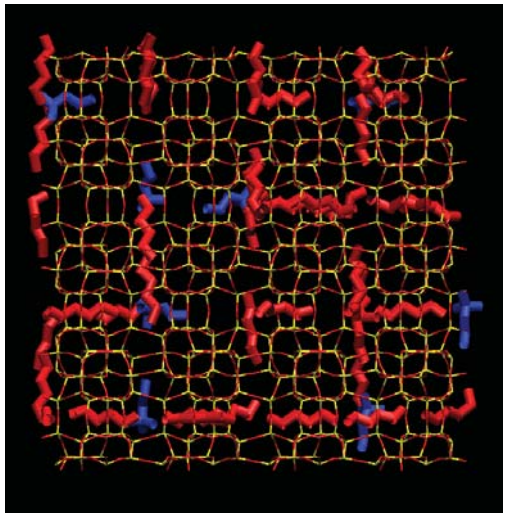
Research Octane Number (RON)



>100 25 73 92

Application 1: Adsorption in MFI-type zeolite (9)

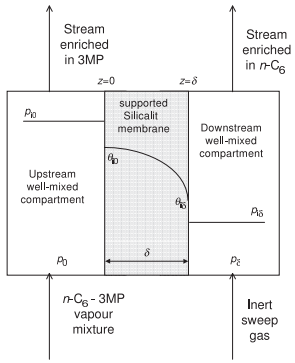
blue = branched (i-C_6) red = linear (n-C_6)



Application 1: Adsorption in MFI-type zeolite (7)

Flux	$n-C_6$	$i-C_6$	selectivity
pure	179	136	1.3
50%-50%	46	1.9	24

Experiments by J. Falconer, Univ. Colorado



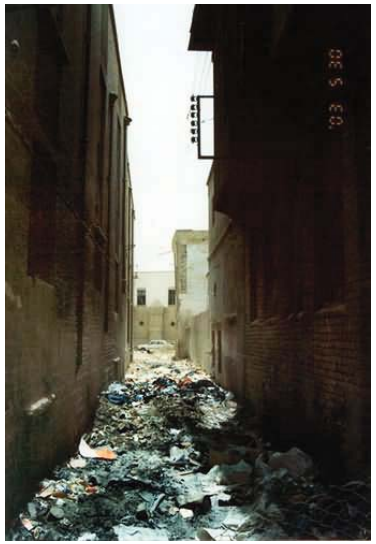
Efficiency of CBMC

- k: number of trial directions
- a: probability that trial direction has a "favorable" energy
- growth can continue as long as at least 1 trial direction is "favorable"
- chain length N

$$P_{\text{success}} = (1-(1-a)^k)^N = \exp[-cN]$$

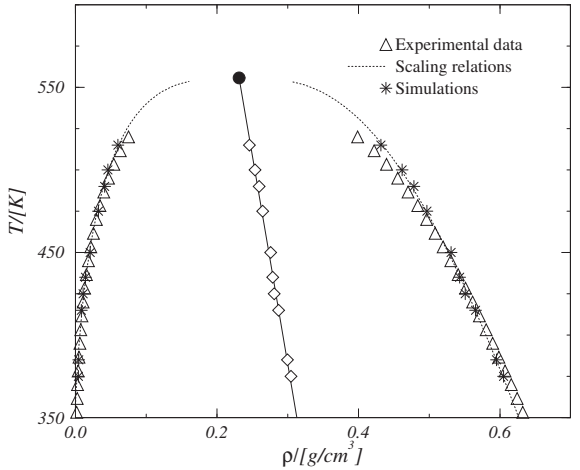
- increasing k means increasing CPU time.

Dead-End Alley



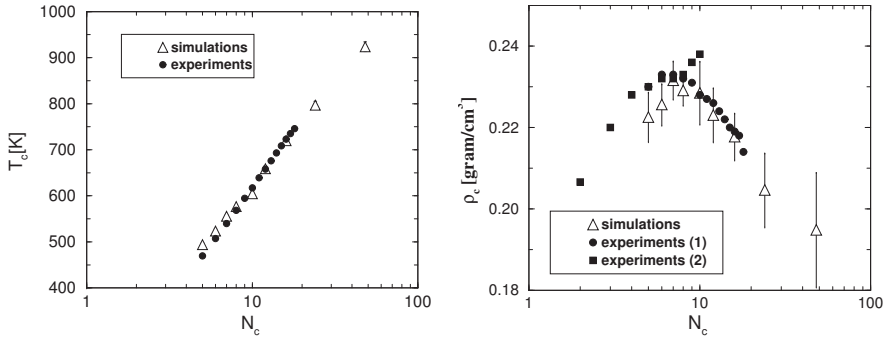
Application 2: Gibbs Ensemble Monte Carlo (1)

B. Smit, S. Karaborni, and J.I. Siepmann, J. Chem. Phys. 102, 2126 (1995)
Heptane (n-C_7)



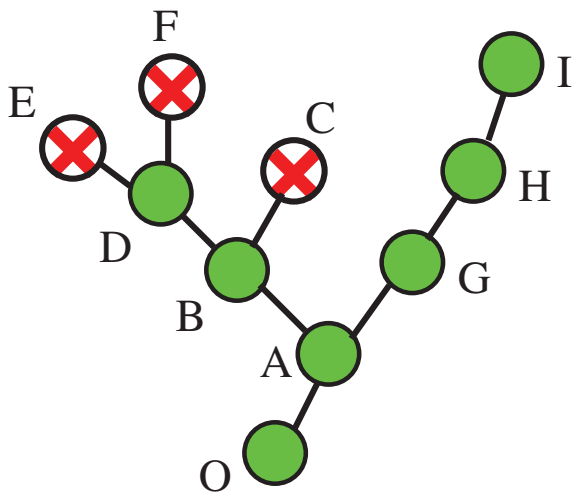
Application 2: Gibbs Ensemble Monte Carlo (2)

B. Smit, S. Karaborni, and J.I. Siepmann, J. Chem. Phys. 102, 2126 (1995)



Recoil Growth Algorithm (2)

Example for $k = 2$ and $l = 3$



Super Detailed Balance

In general,

$$N(o) \times \alpha(o \to n) \times \mathrm{acc}(o \to n) = N(n) \times \alpha(n \to o) \times \mathrm{acc}(n \to o)$$

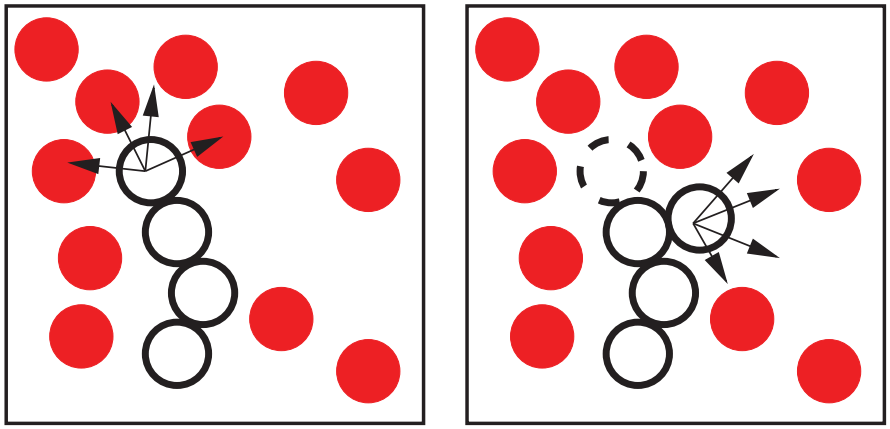
Therefore,

$$\mathrm{acc}(o \to n) = \min\left(1, \exp[-\beta \Delta U] \times \frac{\alpha(n \to o)}{\alpha(o \to n)}\right)$$

What about $\alpha(o \to n)$?

- Generate a tree t_n.
- Decide which parts of the tree are "open" or "closed" (O_n).
- Make a random walk on the tree (rw_n)

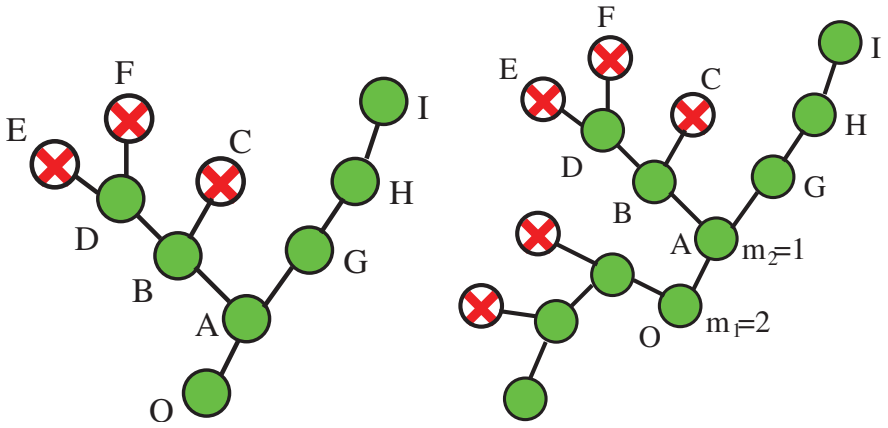
Recoil Growth: Avoiding Dead-End Alley



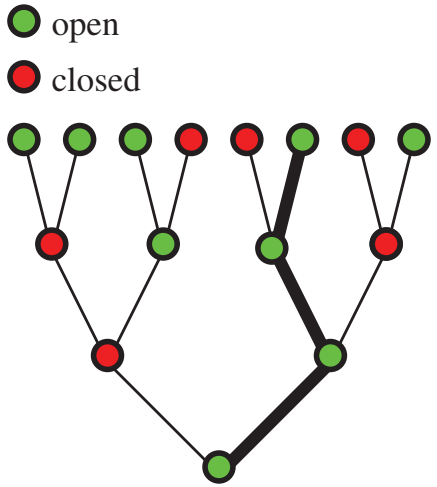
Recoil Growth Algorithm (1)

- Place first bead at random position
- Position (i) can be "open" or closed" depending on the environment (energy u_i); here we use $p_{\mathrm{open}} = \min(1, \exp[-\beta u_i])$ and toss a coin.
- If "open", continue with next segment
- If "closed", try another trial direction up to a maximum of k
- If all k directions are closed, retract by one step
- Maximum retraction length: $l_{\max} - l + 1$
- l: recoil length, $l_{\max}$: maximum length obtained during the growth of the chain
- Computed weight $W(n)$ and repeat procedure for old configuration
- Accept or reject using $\mathrm{acc}(o \to n) = \min(1, W(n)/W(o))$

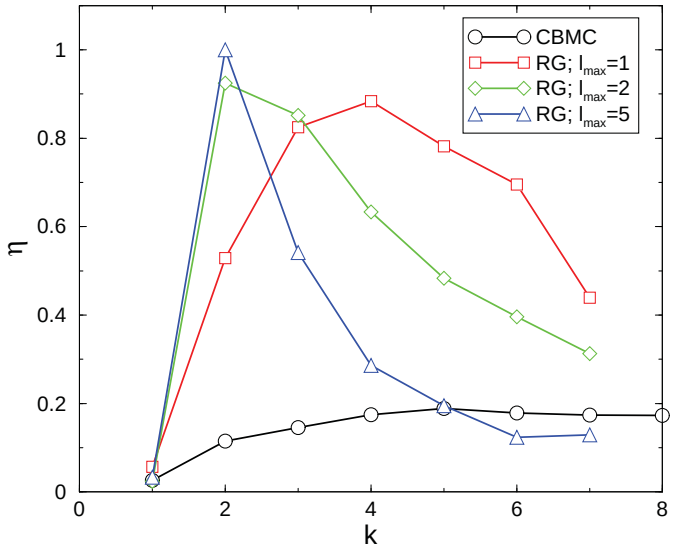
Computing the Weight ($k = 2$, $l = 3$)



Random Walk on a Tree ($k = 2$, $l = 3$)



Efficiency of RG Compared to CBMC



Super-Detailed Balance

$$
\begin{aligned}
K(o \to n|t_n t_o O_n O_o) &= K(n \to o|t_n t_o O_n O_o) \\
\alpha(o \to n|t_n t_o O_n O_o) &= P(t_n)P(O_n|t_n)P(rw_n|t_n O_n) \times \\
&\quad P(t_o|rw_o)P(O_o|t_o rw_o) \\
\alpha(n \to o|t_n t_o O_n O_o) &= P(t_o)P(O_o|t_o)P(rw_o|t_o O_o) \times \\
&\quad P(t_n|rw_n)P(O_n|t_n rw_n) \\
\frac{P(O_n|t_n)}{P(O_n|t_n rw_n)} &= \prod_{i=1}^{n} p_i \\
P(rw_n|t_n O_n) &= \frac{1}{\prod_{i=1}^{n} m_i} \\
\mathrm{acc}(o \to n) &= \min\left(1, \exp[-\beta \Delta U] \times \frac{\prod_{i=1}^{n} \frac{m_i(n)}{p_i(n)}}{\prod_{i=1}^{n} \frac{m_i(o)}{p_i(o)}}\right)
\end{aligned}
$$

The End

Questions ??