Monte Carlo Methods
Parallelization Strategies

Dieter W. Heermann

Heidelberg University

December 7, 2020

Parallelization Strategies

Decomposing the Problem I

The motivation to parallelize the algorithms for the treatment of science problems is manifold:

- It ranges from being able to gather better statistics,
- speeding up an application,
- real time visualization,
- to be able to simulate rather large systems,
- ...

Decomposing the Problem II

In general we can say that there are several types of parallelism inherent in problems. These are:

- Independence: The entities do not interact at all.
- Time correlated: The entities are correlated in time
- Space correlated: The entities are correlated in space

Decomposing the Problem III

One can distinguish between the following general concepts

- Poor man's parallelization
- Data parallelization
- Algorithmic parallelization
- Domain decomposition
- Master-Slave paradigm

These make use of the independence relations that are naturally associated with any problem.

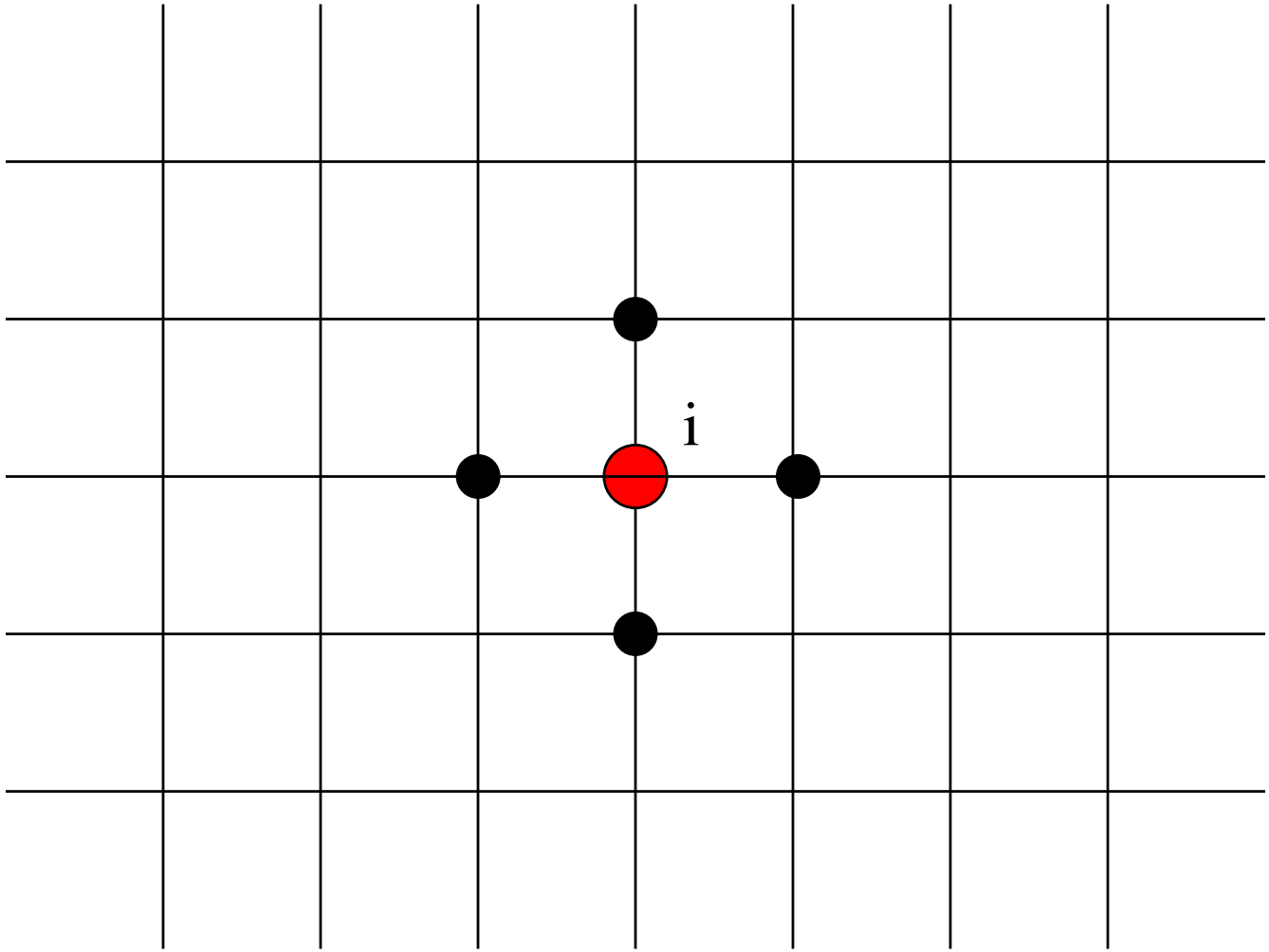
The Ising model I

The Ising model (1) is defined as follows:

- Let $G = L^d$ be a d-dimensional lattice.
- Associated with each lattice site i is a spin s_i which can take on the values $+1$ or -1.
- The spins interact via an exchange coupling J. In addition, we allow for an external field H.
- The Hamiltonian reads

$$\mathcal{H} = -J \sum_{\langle i,j \rangle} s_i s_j + \mu H \sum_i s_i \tag{1}$$

- The first sum on the right-hand side of the equation runs over nearest neighbours only.
- The symbol μ denotes the magnetic moment of a spin. If the exchange constant J is positive, the Hamiltonian is a model for ferromagnetism, i.e., the spins tend to align parallel.
- For J negative the exchange is anti ferromagnetic and the spins tend to align antiparallel. In what follows we assume a ferromagnetic interaction $J > 0$.


i

Metropolis-Hastings Monte Carlo I

- The simplest and most convenient choice for the actual simulation is a transition probability involving only a single spin; all other spins remain fixed.
- It should depend only on the momentary state of the nearest neighbours.
- After all spins have been given the possibility of a flip a new state is created. Symbolically, the *single-spin-flip* transition probability is written as

$$W_i(s_i) : (s_1, ..., s_i, ..., s_N) \longrightarrow (s_1, ..., -s_i, ..., s_N)$$

  where W_i is the probability per unit time that the ith spin changes from s_i to $-s_i$.
- With such a choice the model is called the single-spin-flip Ising model (Glauber).
- Let $P(s)$ be the probability of the state s. In thermal equilibrium at the fixed temperature T and field K, the probability that the i-th spin takes on the value s_i is proportional to the Boltzmann factor

$$P_{eq}(s_i) = \frac{1}{Z} exp\left(\frac{-\mathcal{H}(s_i)}{k_B T}\right)$$

  The fixed spin variables are suppressed.

Metropolis-Hastings Monte Carlo II

- We require that the detailed balance condition be fulfilled:

$$W_i(s_i)P_{eq}(s_i) = W_i(-s_i)P_{eq}(-s_i)$$

or

$$\frac{W_i(s_i)}{W_i(s_i)} = \frac{P_{eq}(-s_i)}{P_{eq}(s_i)}$$

- It follows that

$$\frac{W_i(s_i)}{W_i(s_i)} = \exp(-\Delta\mathcal{H}/k_B T)$$

- The derived conditions do not uniquely specify the transition probability W.
- The **Metropolis function**

$$W_i(s_i) = min\left\{\tau^{-1}, \tau^{-1} exp(-\Delta\mathcal{H}/k_B T)\right\}$$

- and the **Glauber function**

$$W_i(s_i) = \frac{(1 - s_i \tanh E_i/k_B T)}{2\tau}$$

where τ is an arbitrary factor determining the time scale.

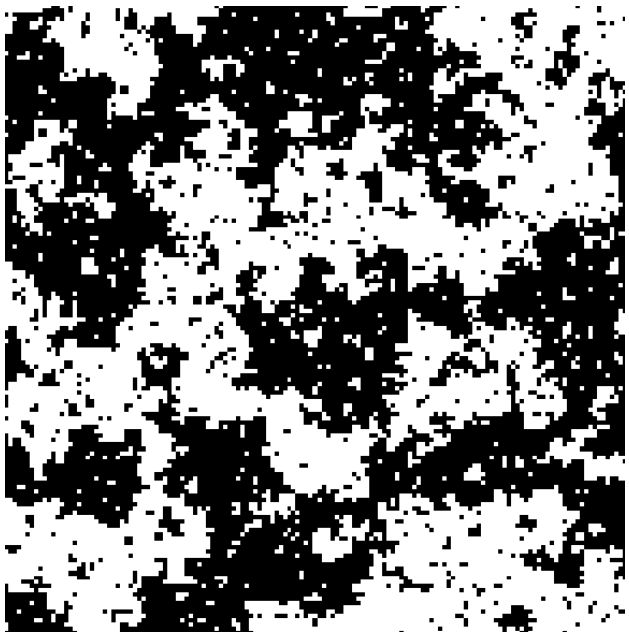
Algorithm: Metropolis-Hastings Monte Carlo

Algorithmically the Metropolis MC method looks as follows:

Algorithm 1 Metropolis-Hastings Monte Carlo Method

1: Specify an initial configuration.
2: Choose a lattice site i.
3: Compute W_i.
4: Generate a random number $R \in [0, 1]$.
5: **if** $W_i(s_i) > R$ **then**
6: $\quad s_i \rightarrow -s_i$
7: **else**
8: $\quad$ Otherwise, proceed with Step 2 until *MCSMAX* attempts have been made.
9: **end if**

Program (Java): Ising Model MC



Java program can be found here

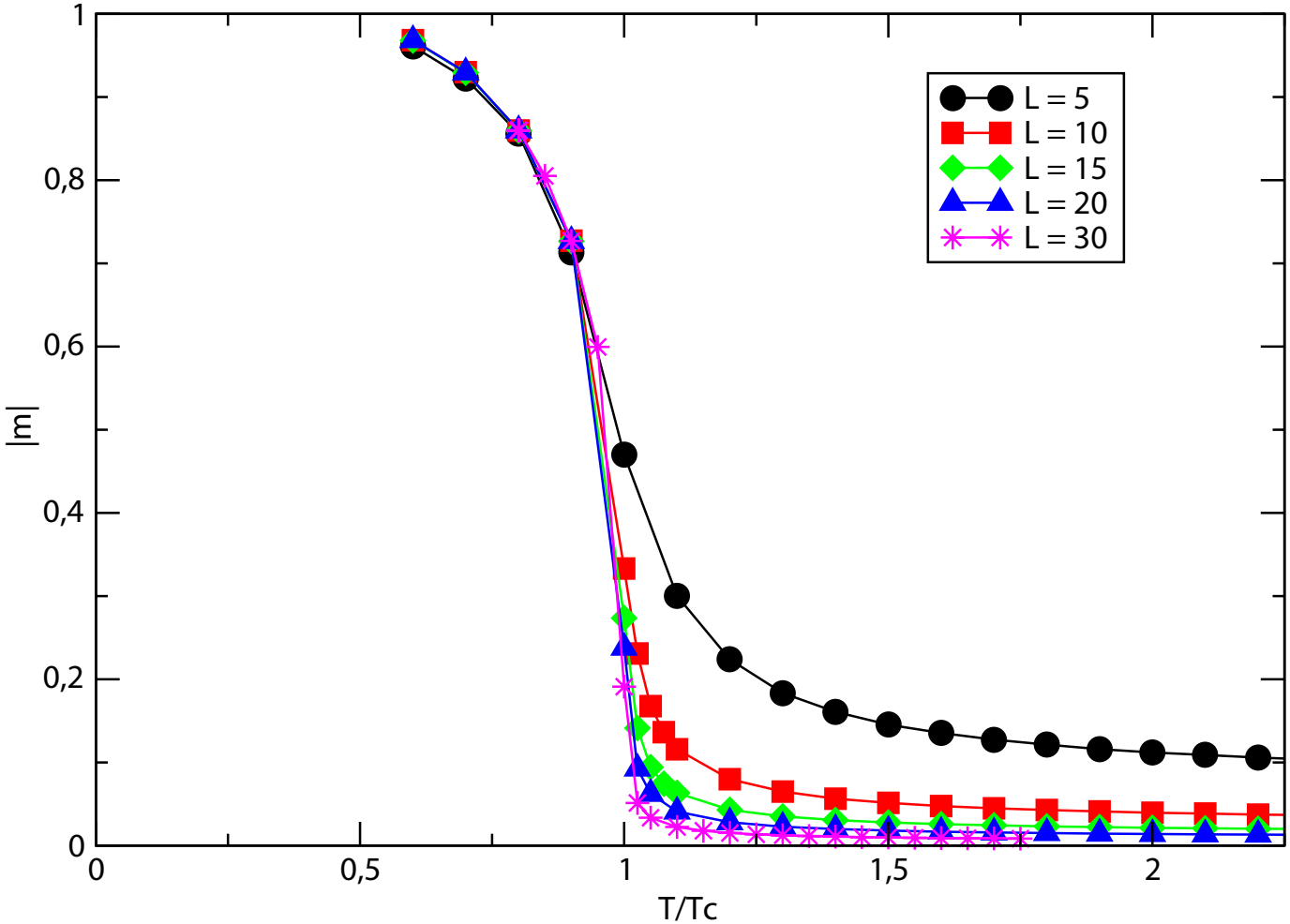
Program (C): Ising Model MC

The C program can be found here

```
#include <iostream.h>
#include <math.h>

# define L  10

int    main(int argc, char *argv[])
{
    int mcs,i,j,k,ip,jp,kp,in,jn,kn;
    int old_spin,new_spin,spin_sum;
    int old_energy,new_energy;
    int mcsmax;
    int spin[L][L][L];
    int seed;
    double r;
    double beta;
    double energy_diff;
    double mag;

 mcsmax = 100;
 beta = 0.12;       // beta = J/kT  KC = 0.2216544 Talapov and Blöte (1996)
 seed = 4711;

 srand(seed);
    for (i=0;i<L;i++) {
        for (j=0;j<L;j++) {
            for (k=0;k<L;k++) {

             spin[i][j][k] = -1;
            }
        }
    }
 mag = - L*L*L;

    // Loop over sweeps
    for (mcs=0;mcs<mcsmax;mcs++) {

        // Loop over all sites
        for (i=0;i<L;i++) {
            for (j=0;j<L;j++) {
                for (k=0;k<L;k++) {

                   // periodic boundary conditions
                 ip = (i+1) % L;
                 jp = (j+1) % L;
                 kp = (k+1) % L;
                 in = (i+L-1) % L;
                 jn = (j+L-1) % L;
                 kn = (k+L-1) % L;

                 old_spin = spin[i][j][k];
                 new_spin = - old_spin;

                   // Sum of neighboring spins
                 spin_sum = spin[i][jp][k] + spin[ip][j][k] +
                 spin[i][jn][k] + spin[in][j][k] +
                 spin[i][j][kn] + spin[i][j][kp];

                 old_energy = - old_spin * spin_sum;
                 new_energy = - new_spin * spin_sum;
```

3D Ising Model

Magnetization vs. Temperature



Ising Model MC (continued)

The Ising model can also be studied using the formulation of a random cluster model (5) with the partition function

$$\mathcal{Z} = \sum_{C} B(\beta, C) 2^{n(C)} \quad . \tag{2}$$

- In this formulation the clusters are independent but stretch over the entire lattice. We have lost the locality inherent in the original formulation.
- This implies that the lattice does not partition into independent sublattices!
- The formulation of the model can play a central role whether one can parallelize a problem to a good degree

This does imply that the better parallelized problem is not necessarily more efficient!

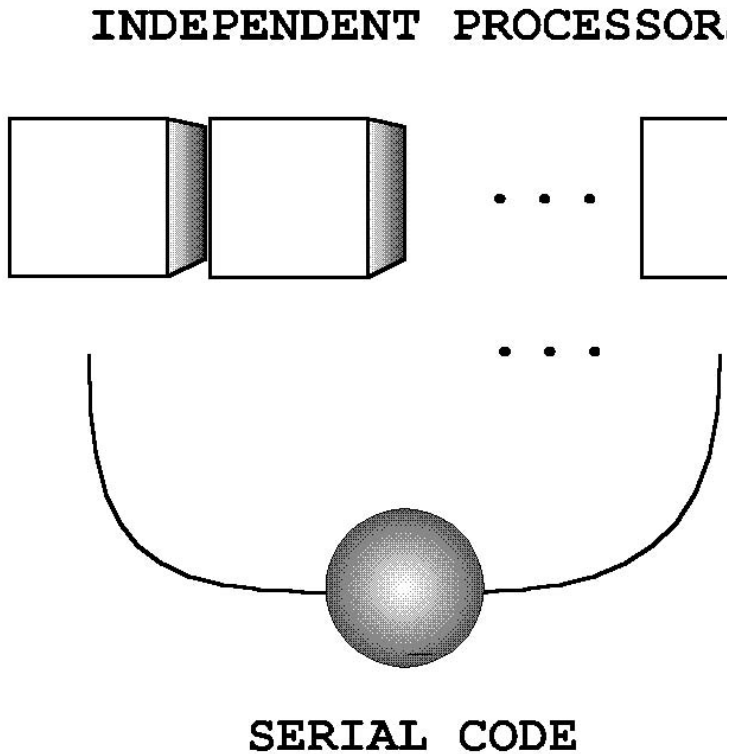
We should not confuse efficiency and effectiveness

Embarrassingly Parallelizable Monte Carlo

Today, the most often used parallelization is presumably the so called poor man's parallelization or also called embarrassingly parallelizable problem
An application is **embarrassingly parallel** if its parallel implementation

- can straightforwardly be broken up into roughly equal amounts of work per processor,
- has minimal communication overhead among processors.
- In the poor man's parallelization a given code is replicated as many times as there are processors.
- Each program on a processor executes independently from the other copies.
- No communication beside the input/output of results is necessary, that is, the programs or processes on different processors do not communicate with each other.
- The efficiency with which the processors are used is 100 percent.

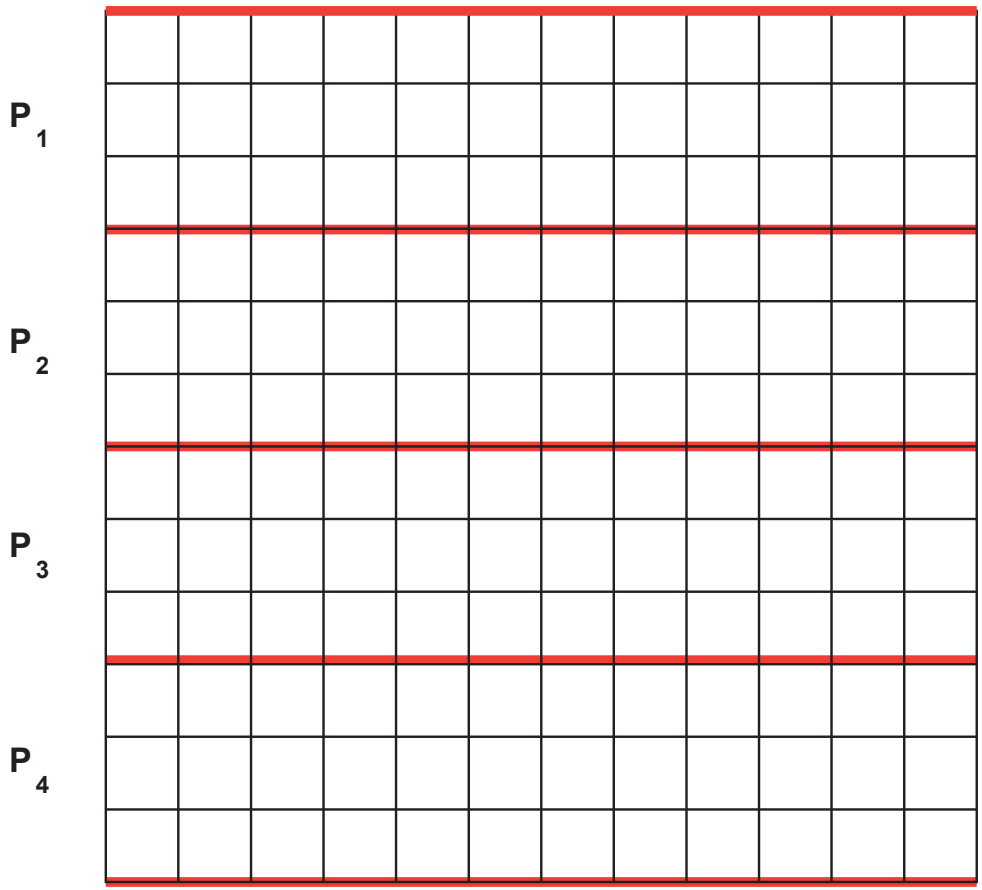
For the gathering of statistics in Monte Carlo problems, this concept is very well suited.



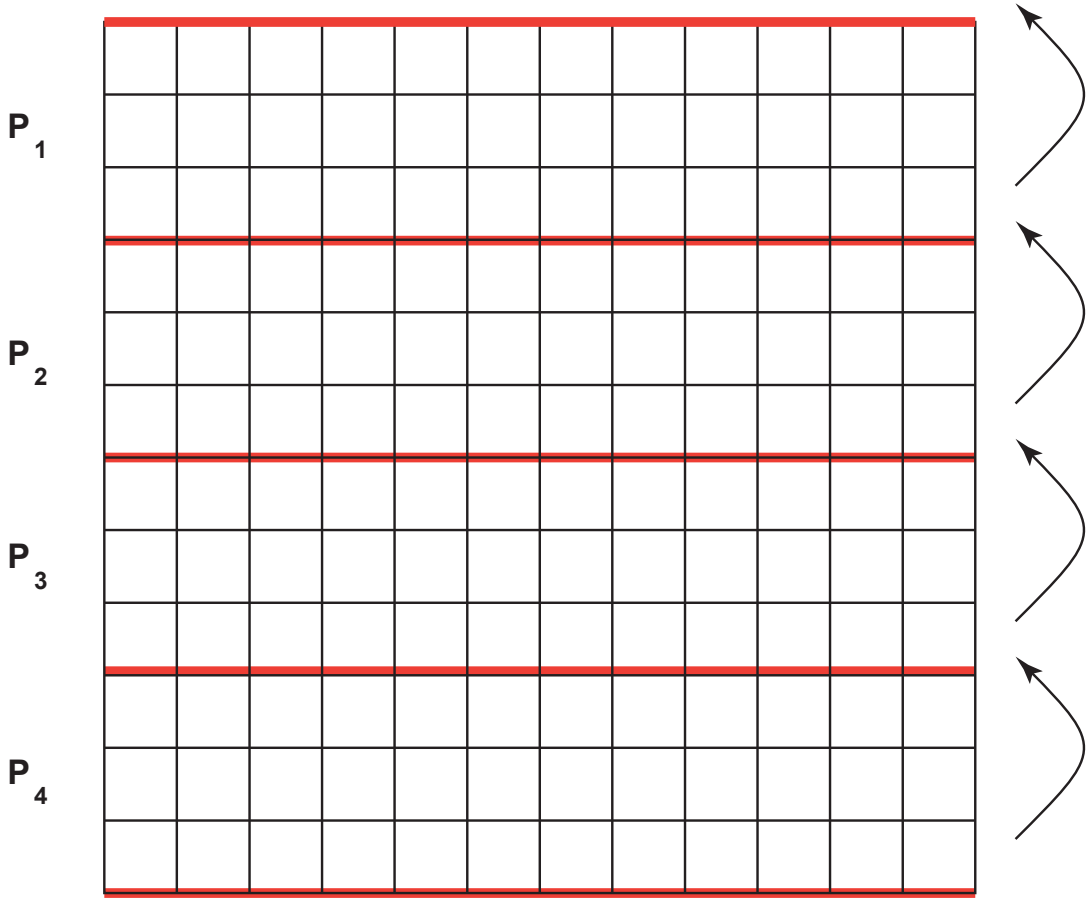
In the poor mans parallelization a production code for a single processor machine is distributed in identical copies to the number of available or desired processors. All execute concurrently without communication among the copies.

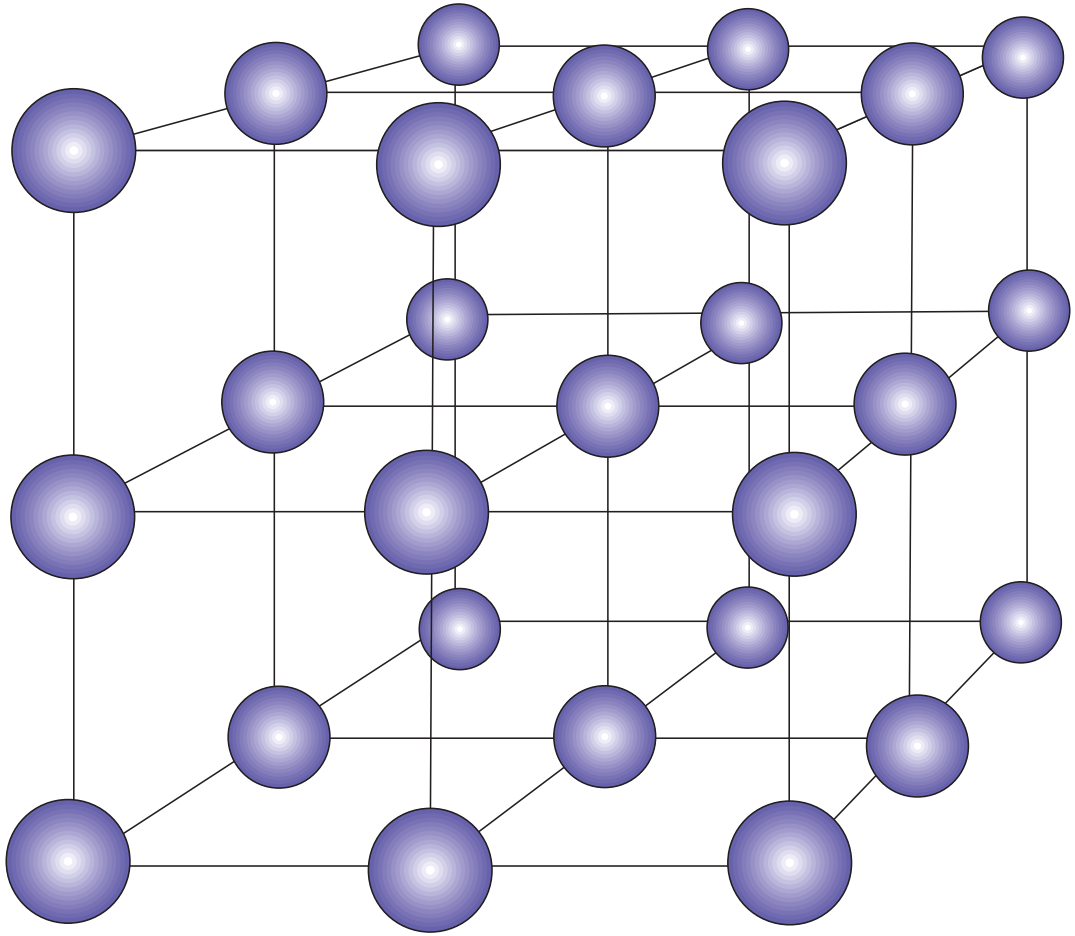
Geometric (Domain) Decomposition

Divide the computational domain into n equally-sized sub-domains.
Strip Decomposition

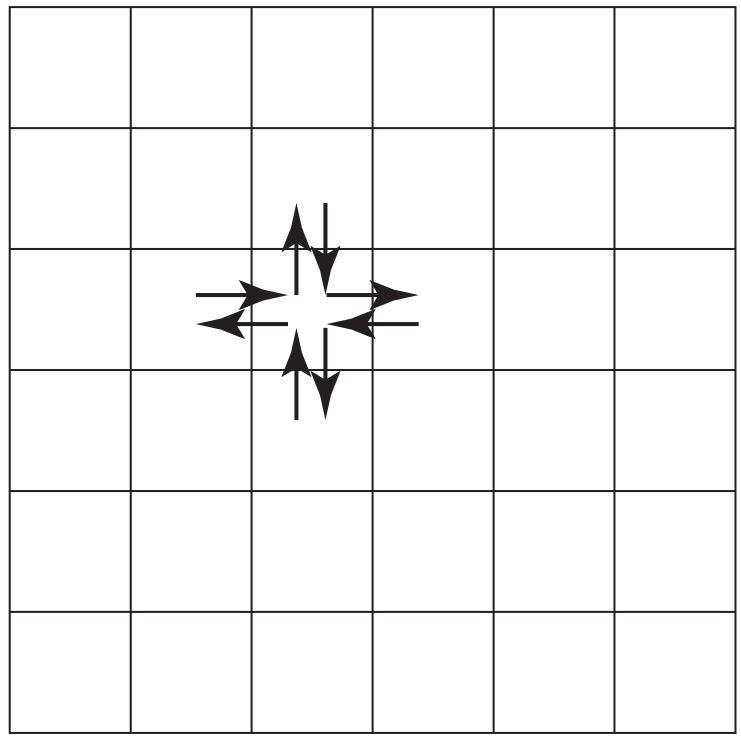


Usually we have to take into account periodic boundary conditions



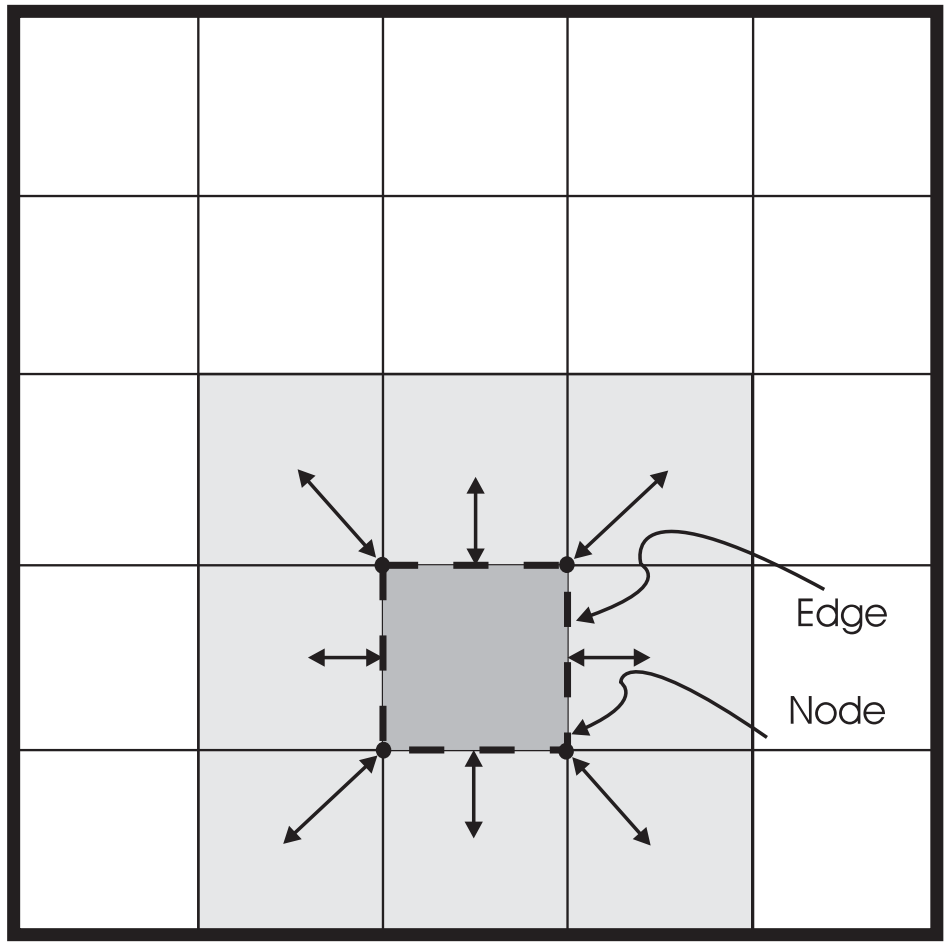


Typically, each site in a multi-dimensional geometry is updated with contributio
from a subset of its neighbors (**Stencil computations**)



One must ensure at all times that processors owning neighboring sub-domains do not update adjacent sites simultaneously. (detailed balance)

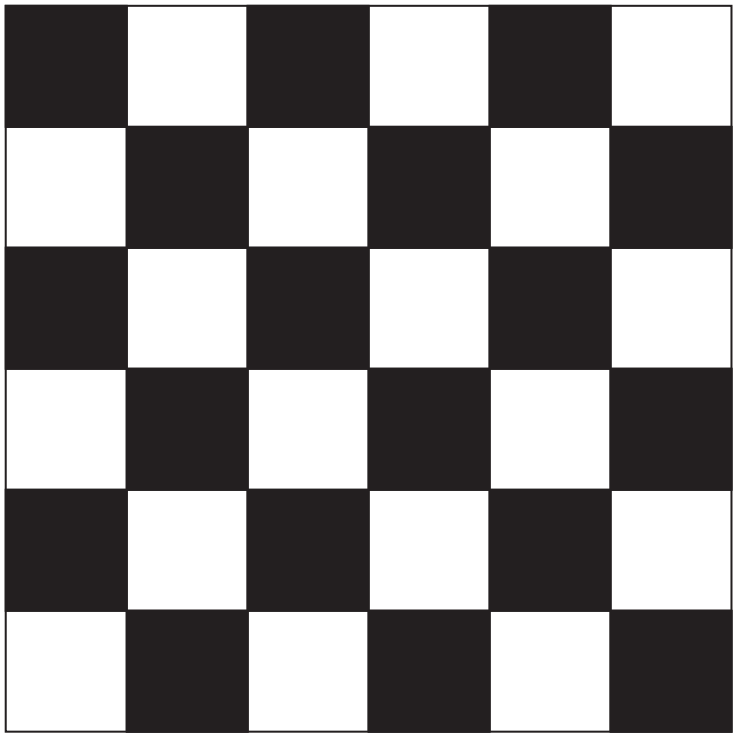
The message passing paradigm provides a simple way to implement the lattice updates without violating data dependencies.



- The efficiency of geometric decomposition methods is strongly affected by heterogeneity and variability present in the underlying hardware.
- The processors are tightly coupled by the communication phases.
- Due to the Monte Carlo method (and due to load fluctuations in a domain) the execution time is constraint by the slowest processor.
- SMP architecture may effect execution time due to competition for memory resources.

Vectorization

Example: Vectorizing the Ising Model
Use the checker board idea to ensure that detailed balance is fulfilled (see (7))



Vectorization

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Vectorized Ising Model run on the FACOM VP-100 Series (see Historical computers in Japan)

```
c
c--------2dvp100-----------------------------------------------------------
c
c   program      : monte carlo of the two dimensional ising model
c
c   algorithm    : based on the skewed numbering of the two sub-
c                  lattices vectorization is possible (w. oed, angw.
c                  inf. 7,358 (1982).
c
c   machine      : vp100
c
c   remarks      : the lattice dimension l must always by a
c                  multiple of 2
c
c   author       : dieter w. heermann
c
c   version 1.0    december 1986
c
c   last test    : 13/3/87
c
c-----------------------------------------------------------------------
c
      dimension subl1(1:465),subl2(1:465)
      dimension vec1(1:465),vec3(1:465)
      dimension rvec(1:450)
      dimension ranf(1:450)
      dimension tb(-4:4)
c
      integer   subl1,subl2
      integer   vec1,vec3
      integer   vl,vlp1,lh
      integer   offs1,offs2
c
      real      tb
      real      jkt,ttc
c
c             simulation parameters
c             =====================
c
      l      = 10
      read(5,*) l
      maxmcs = 300
      ttc    = 0.95
      read(5,*) ttc
      iseed1 = 51
      iseed2 = 514
      read(5,*) iseed1
      read(5,*) iseed2
c
      write(6,*) 'system size is    = ',l
      write(6,*) 'temperature is    = ',ttc
      write(6,*) 'monte carlo steps = ',ttc

c
c             set up parameters
c             =================
c
      jkt    = 0.4406867935 / ttc
      vlp1   = (l + 1) * l / 2
      vl     = l * l  / 2
      lh     = l/2
```

- The master initially distributes one task to every slave.
- The slaves compute their tasks and send the results back to the master,
- Each slave triggers the master to send additional tasks.
- This is self-scheduling, demand-driven or first-come first-served (FCFS).
- FCFS is not efficient when point-to- point communication times are heterogeneous.

Conclusions

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

- Monte Carlo algorithms can be converted from serial to parallel algorithms.
- Vectorization can often be achieved.
- Random number algorithms are very important!

[1] E. Ising: Z. Phys. 31,253 (1925)

[2] D.W. Heermann and A.N. Burkitt, **Parallel Algorithms of Computational Science Problems** Springer Verlag, Heidelberg 1990

[3] K. Binder and D.W. Heermann, **The Monte Carlo Method in Statistical Physics: An Introduction** Springer Verlag, Heidelberg 1988

[4] D.W. Heermann, **Computer Simulation Methods in Theoretical Physics**, Springer Verlag, Heidelberg 1986

[5] C.M. Fortuin and P.W. Kastelyn *Physica* **50**, 297 (1972)

[6] W. Janke and R. Villanova. Ising model on three-dimensional random lattices: A Monte Carlo study. *Physical Review B* , 66(13):134208 (2002)

[7] W. Oed: Angew. Inf. 7/82, 358 (1982)