Scientific Computing Methods for Plasma Physics Simulation

Hans De Sterck

desterck@colorado.edu

Department of Applied Mathematics, University of Colorado at Boulder, USA

(http://amath.colorado.edu/faculty/desterck)

Overview

(1) Numerical Simulation of MHD Bow Shock Flows

(2) $\nabla \cdot \vec{B} = 0$: Constrained Transport on Unstructured Grids

(3) Java Taskspaces for Grid Computing

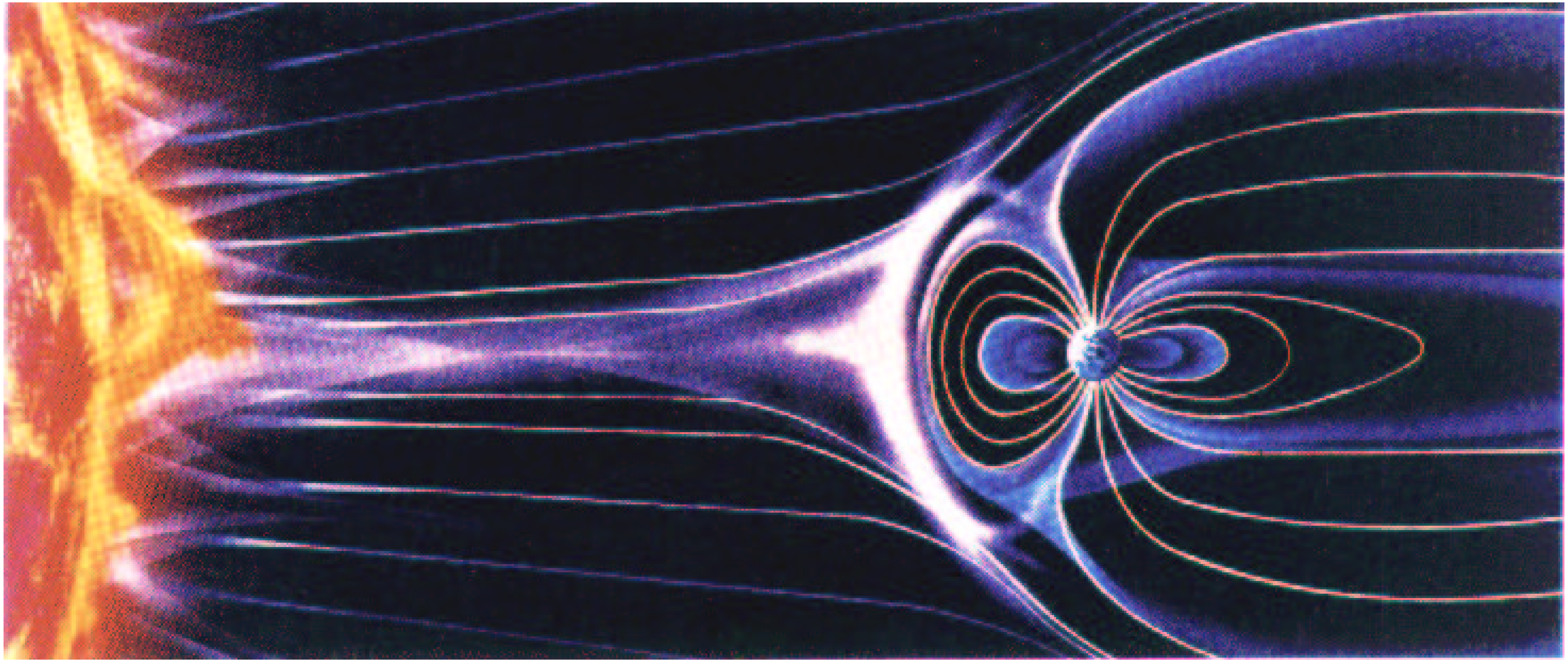
(1) Numerical Simulation of MHD Bow Shock Flows

Conservative form ideal MHD equations

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho\vec{v}\\ \rho e\\ \vec{B}\end{bmatrix} + \nabla\cdot\begin{bmatrix}\rho\vec{v}\\ \rho\vec{v}\vec{v}+\left(p+\frac{B^2}{2}\right)\vec{I}-\vec{B}\vec{B}\\ \left(\rho e+p+\frac{B^2}{2}\right)\vec{v}-(\vec{v}\cdot\vec{B})\vec{B}\\ \vec{v}\vec{B}-\vec{B}\vec{v}\end{bmatrix} = 0$$

- nonlinear system of hyperbolic conservation laws describing magnetized fluid
- applies to laboratory and space plasmas
- constraint: $\nabla\cdot\vec{B}=0$

MHD in Space Physics flows



- (quasi-) steady: supersonic solar wind, magnetosphere
- unsteady flow : coronal mass ejections, magnetic storms
- continuous flow (waves) and discontinuities (shocks)

nonlinear hyperbolic system: waves and shocks

- HD (Euler): ($n = 5$)
  - $\lambda = u, u, u, u + c, u - c$
  - one nonlinear wave mode
  - isotropic
  - one type of shock

- MHD: ($n = 8$)
  - $\lambda = u, u, u + c_f, u - c_f,$
    $u + c_A, u - c_A, u + c_s, u - c_s$
  - three wave modes: fast, Alfven, slow
  - strongly anisotropic
  - three types of shocks

- hyperbolic theory of MHD:
  - non-strictly hyperbolic
  - non-convex $\Rightarrow$ compound shocks
  - rotationally invariant $\Rightarrow$ instability of (overcompressive) intermediate shocks

Numerical simulation technique

- MHD is hyperbolic, like Euler $\Rightarrow$ use CFD techniques
- finite volume, structured grid
- second order in space (limited slope reconstruction)
- second order in time (explicit two-stage Runge-Kutta)
- parallel using message passing (MPI)
- $\nabla \cdot \vec{B}$ constraint:

  MHD has singular Jacobians, which leads to numerical instabilities

  $\Rightarrow$ add source term (K. Powell)

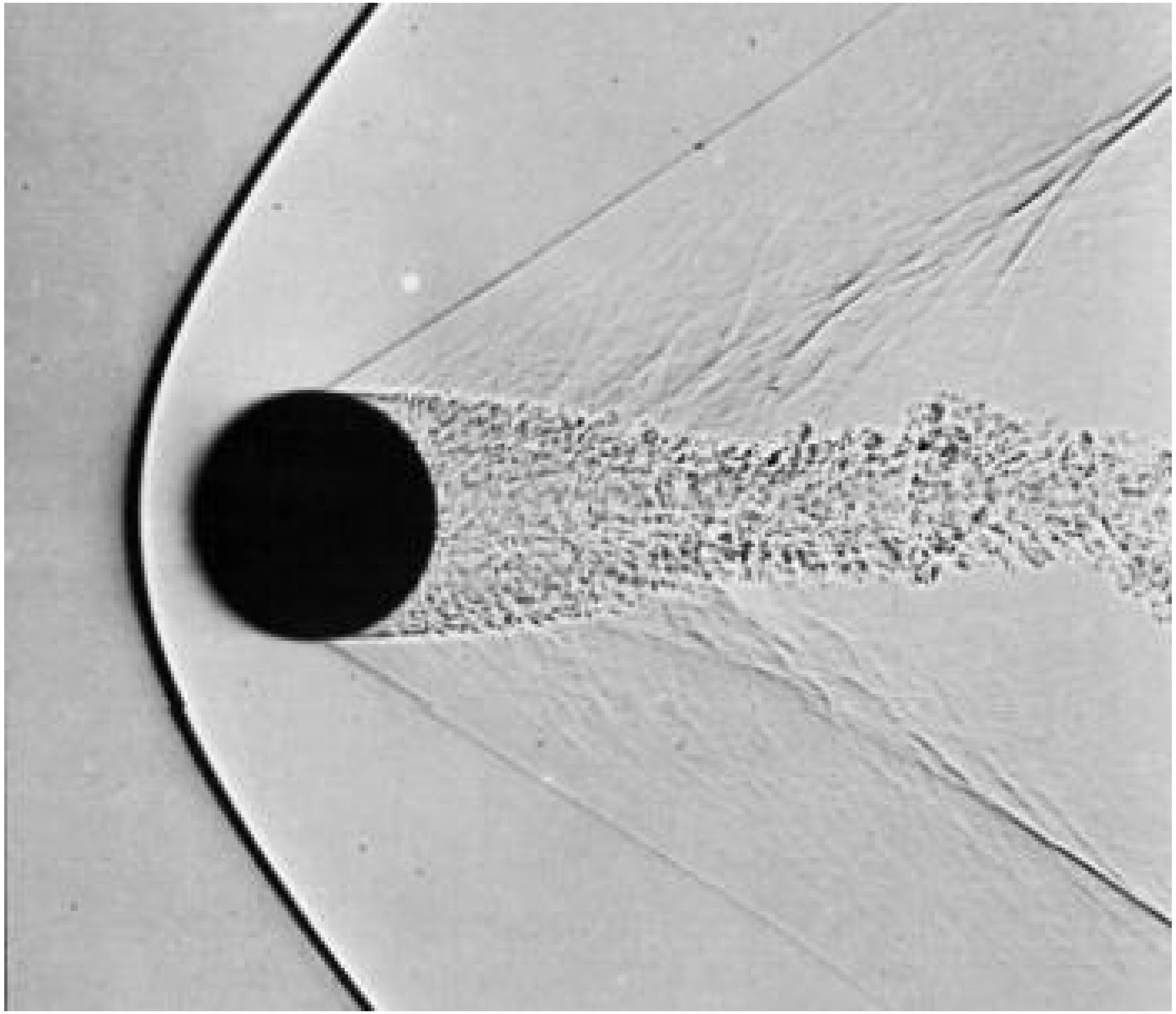
  - makes equations symmetrizable

  - makes equations Galilean invariant

  - makes numerical scheme stable

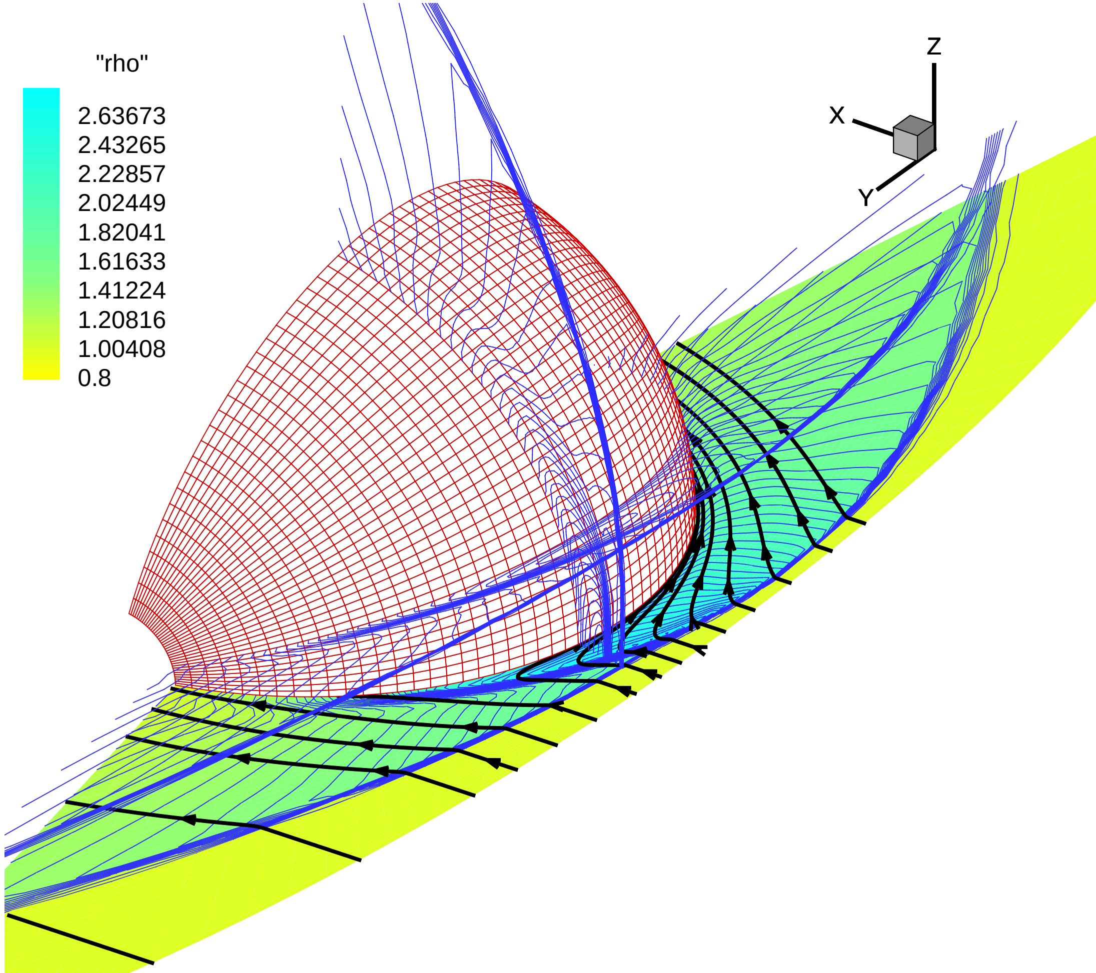
  (alternative techniques: D. Kroener, C.-D. Munz, part (2) of this talk)

(1) Numerical Simulation of MHD Bow Shock Flows



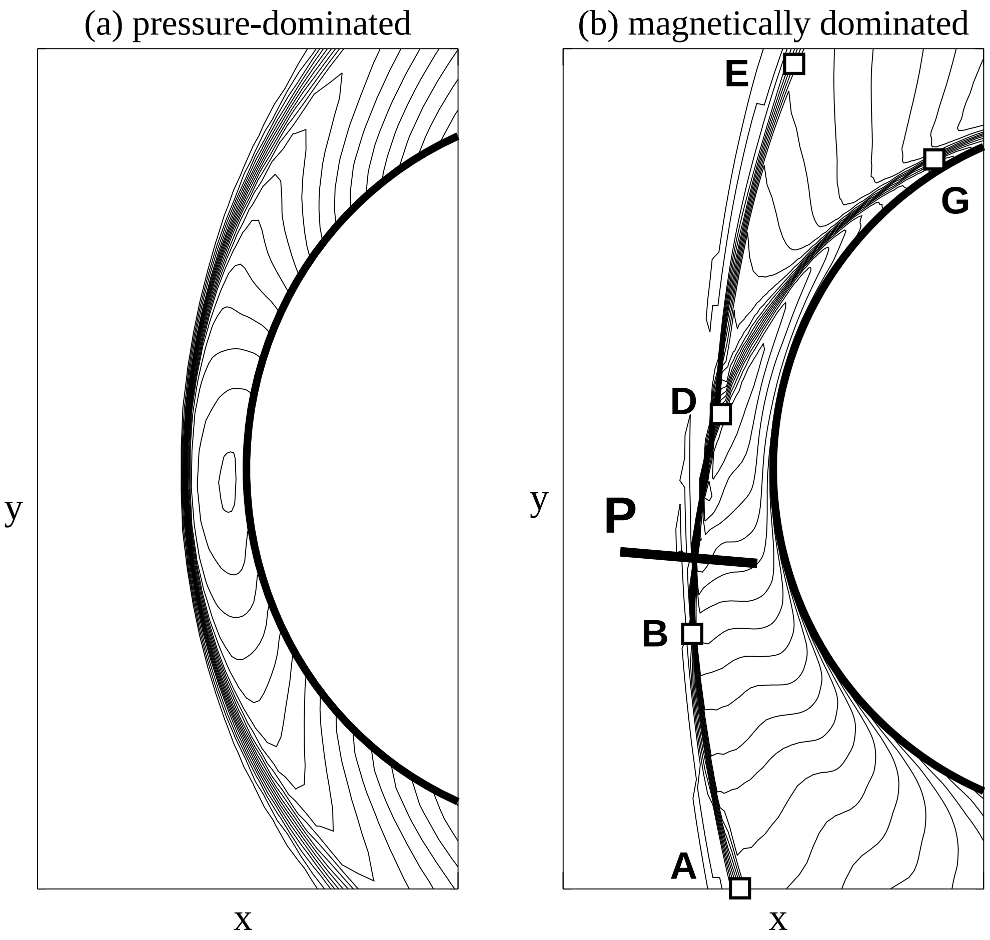
- supersonic flow of air over sphere (M=1.53)
- regular bow shock
- (An album of fluid motion, Van Dyke)

3D MHD simulations: flow over conducting sphere



- finite volume method
- shock-capturing

3D bow shock: complex nonlinear wave structure



B weak B strong

- double-shock topology arises for strong upstream magnetic field
- flow exhibits compound shock
- flow exhibits intermediate overcompressive shock
- some observational evidence for intermediate shocks in CME bow shocks (Steinolfson and Hundhausen, JGR, 1990) and in Venus bow shock (Kivelson et. al., Science, 1991)
- De Sterck and Poedts, JGR 1999, 2001; Phys. Rev. Lett. 2000

Non-convexity: compound shocks

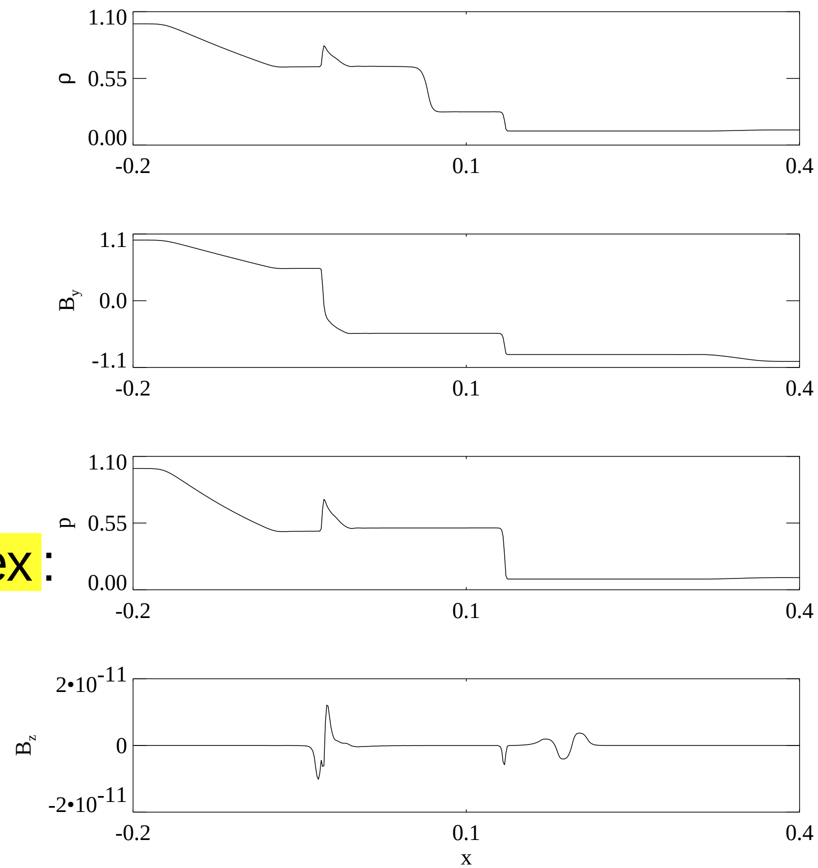
$$\frac{\partial u}{\partial t} + f'(u)\frac{\partial u}{\partial x} = 0$$

f is nonconvex $\Leftrightarrow$ f' is not monotone $\Leftrightarrow$ f'' does change sign

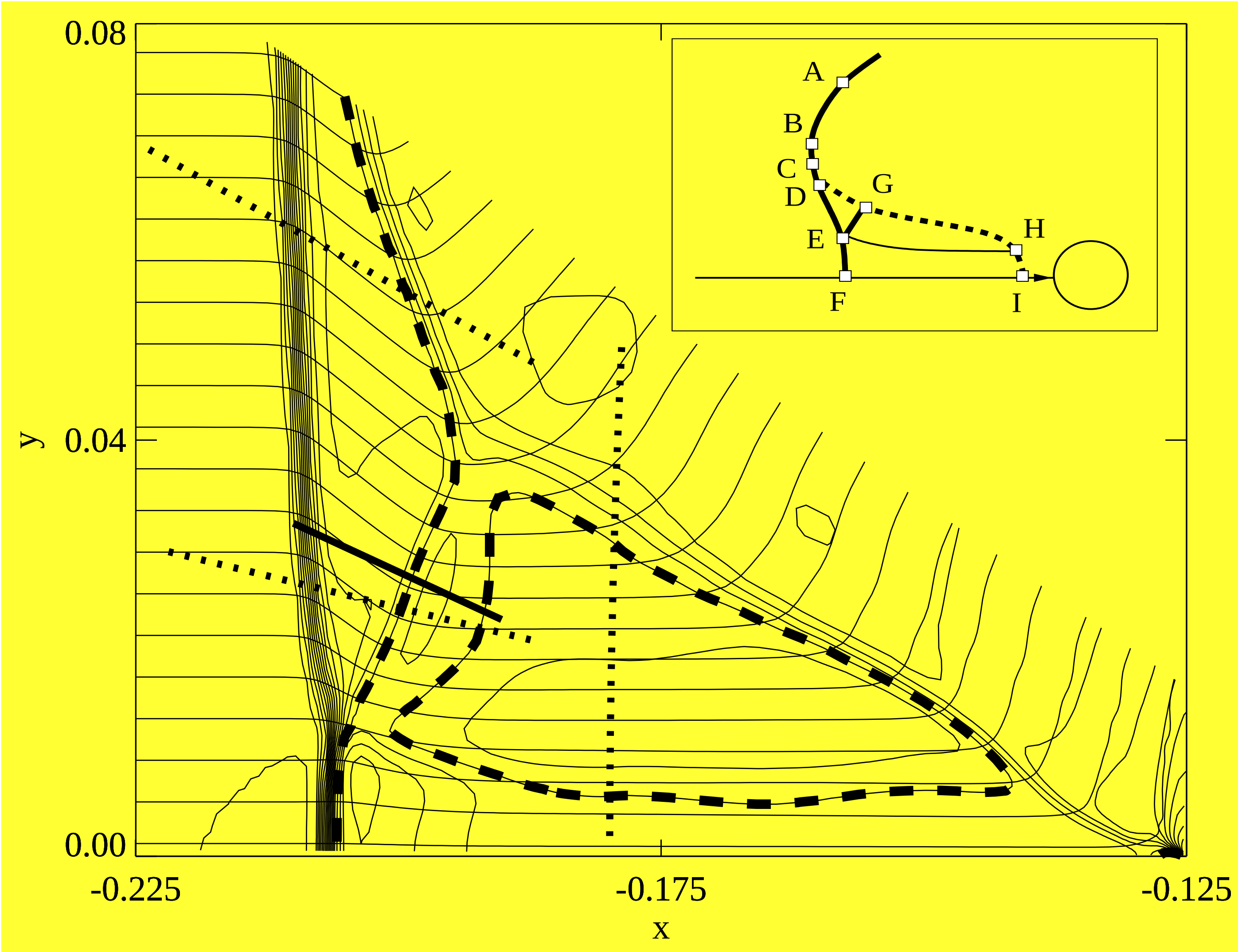
$\Rightarrow$ a compound shock may arise: shock with attached rarefaction which move at same speed

- Euler: all waves are convex
- Euler + combustion: compound shocks
- MHD: fast and slow waves are non-convex: compound shocks!

(this was known in 1D: Brio and Wu, JCP 1988)

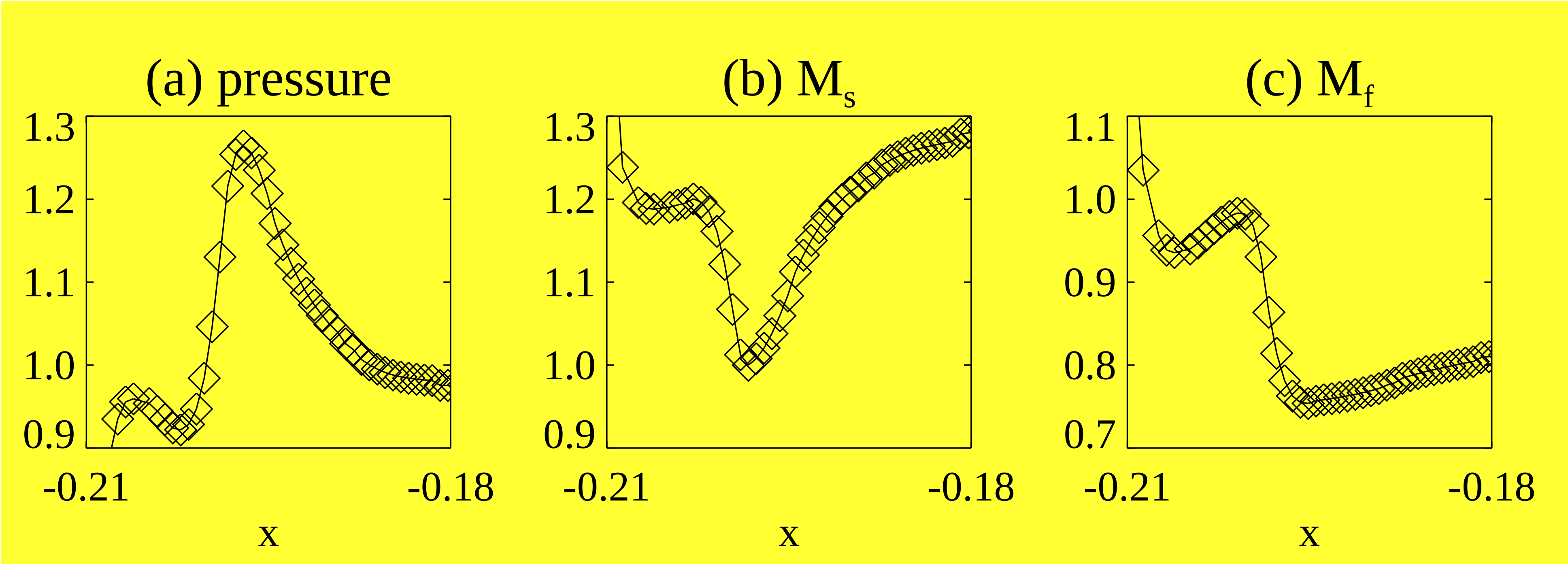


compound shocks in 2D MHD flow



Magnetic field lines and M_A contour lines

(1) Numerical Simulation of MHD Bow Shock Flows



Cut along solid line

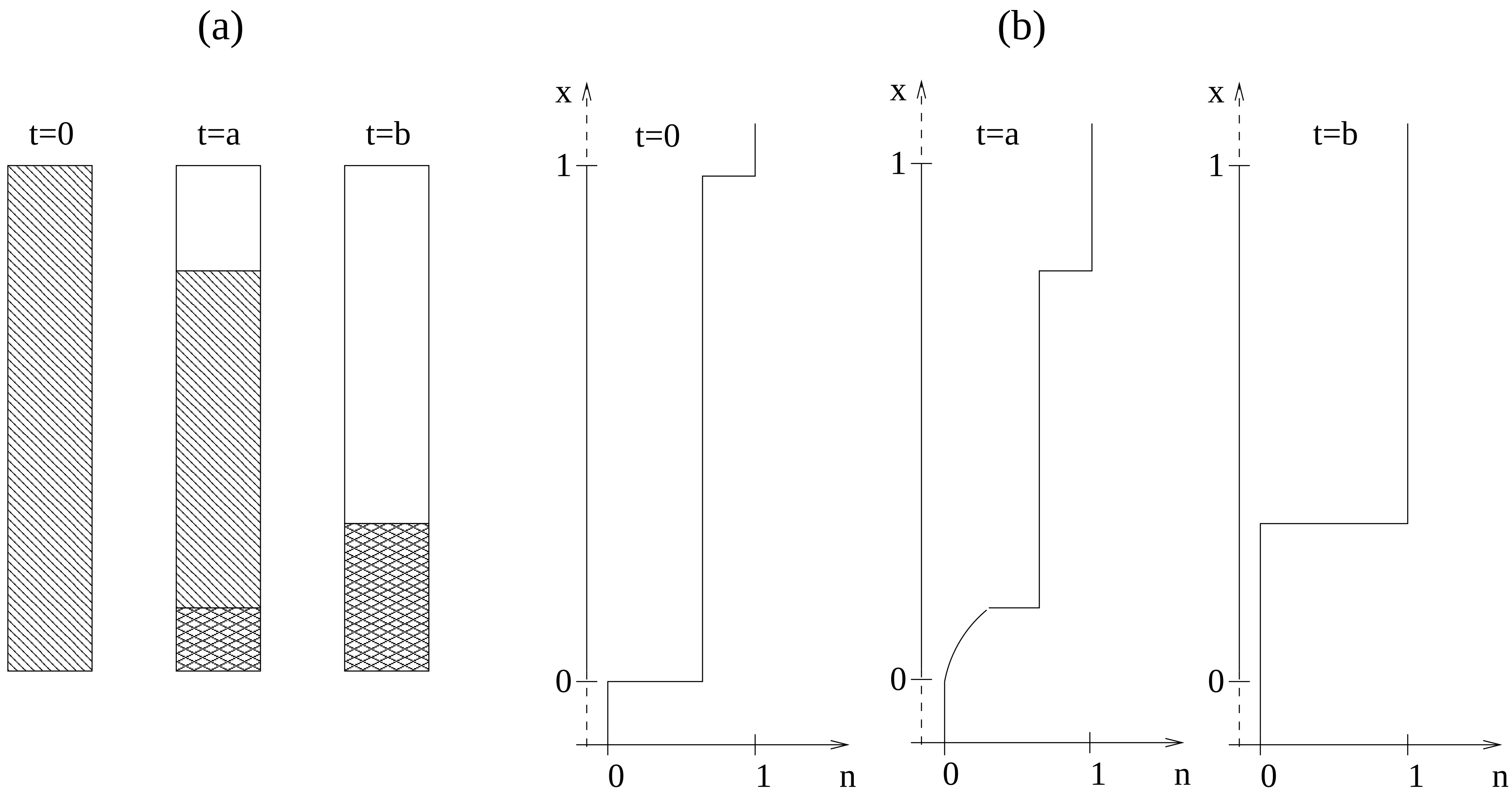
- E-G shock is preceded and followed by rarefaction regions
- $M_f = 1$ where upstream (left) rarefaction is attached to shock
- $M_s = 1$ where downstream (right) rarefaction is attached to shock

$\Rightarrow$ E-G: 1=2–3=4 shock

$\Rightarrow$ stationary double compound shock!   (also in 3D)

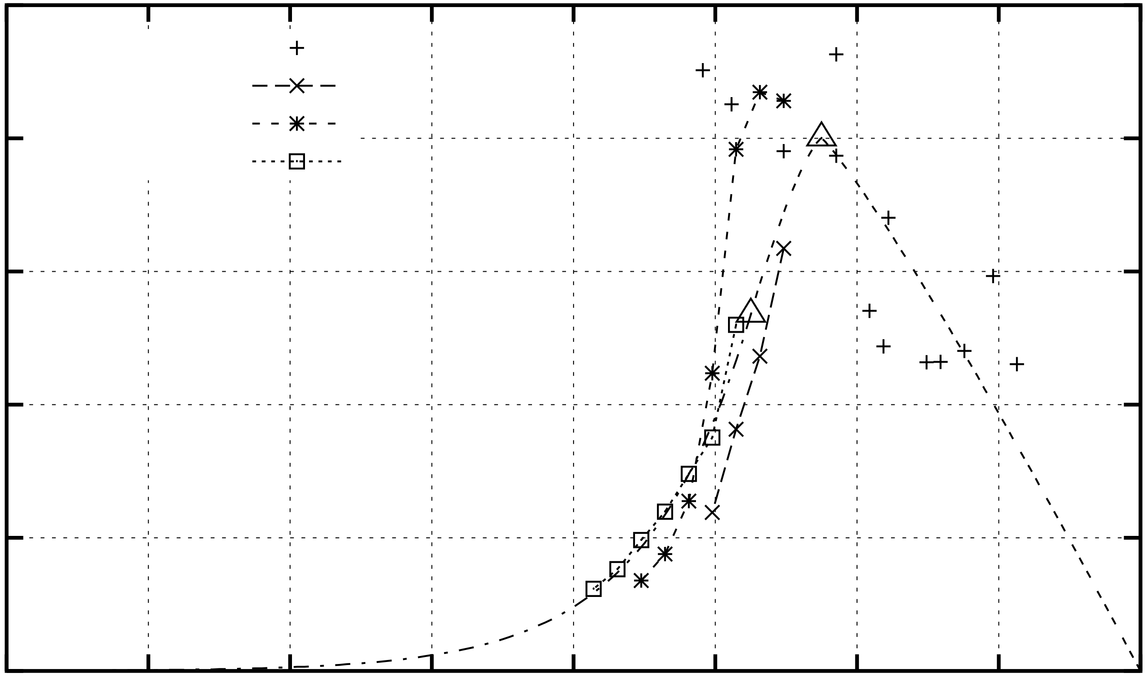
Compound Shock Waves in Sediment Beds

- soil sedimentation experiments in a settling column
- with G. Bartholomeeusen (University of Oxford)

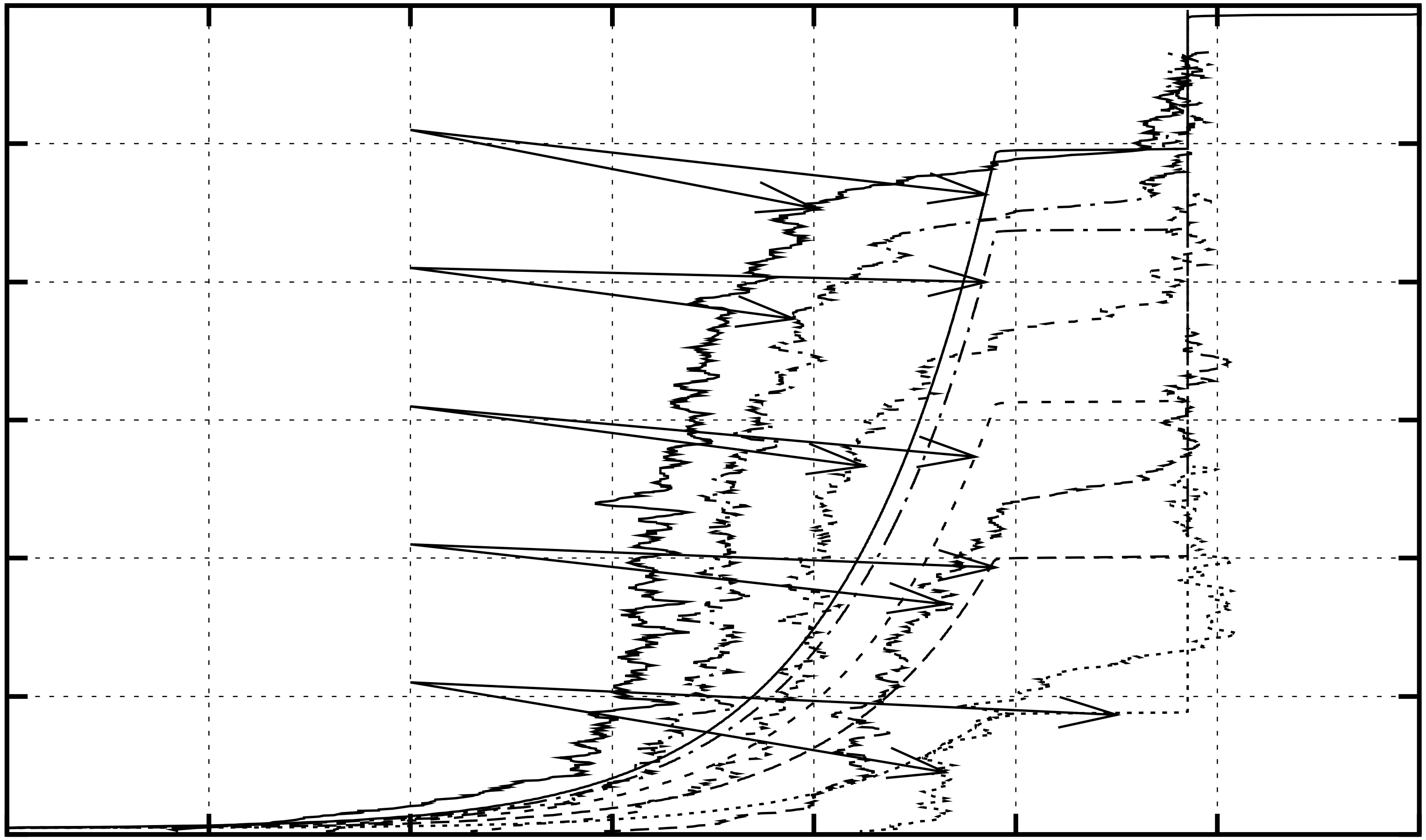


(1) Numerical Simulation of MHD Bow Shock Flows

- nonconvex flux function
- compound shock experimentally observed!
- numerically modeled using experimentally obtained flux function
- submitted to HYP2002
- remark: compound shocks also in oil recovery problems



(1) Numerical Simulation of MHD Bow Shock Flows



(2) Multi-Dimensional Upwind Constrained Transport

MUCT = Multi-Dimensional Upwind Constrained Transport

- **numerical schemes for the advection of divergence-free fields on unstructured grids**

⇒ divergence-free: $\nabla \cdot \vec{B} = 0 \qquad (\text{or} \quad \oint \vec{B} \cdot \vec{n} dS = 0)$

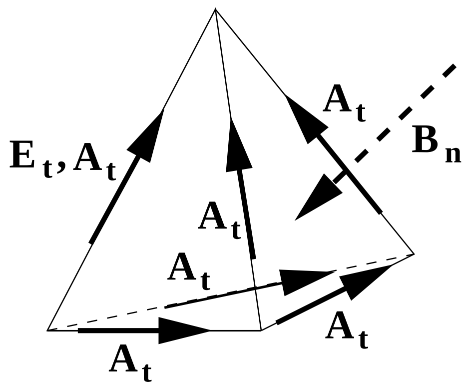
- $\vec{B}$ magnetic field (plasma ...)
- no magnetic monopoles
- also numerically, avoid magnetic monopoles at the discrete level:

  Constrained Transport (CT) approach

⇒ CT was known on structured grids (Evans & Hawley 1988, earlier for EM)

⇒ De Sterck, AIAA CFD paper 2001-2623: how to do constrained transport on unstructured grids

CT: general idea



Faraday: $\frac{\partial \vec{B}}{\partial t} = \nabla \times (\vec{v} \times \vec{B})$

(2) $$\frac{\partial \int \vec{B} \cdot \vec{n} dS}{\partial t} = \oint (\vec{v} \times \vec{B}) \cdot d\vec{l}$$

$$\int \vec{B} \cdot \vec{n} dS = \bar{B}_n \Delta S \quad \Rightarrow \quad \frac{\partial \bar{B}_n}{\partial t} = \oint (\vec{v} \times \vec{B}) \cdot d\vec{l} \, / \, \Delta S$$

= time evolution of flux through surface

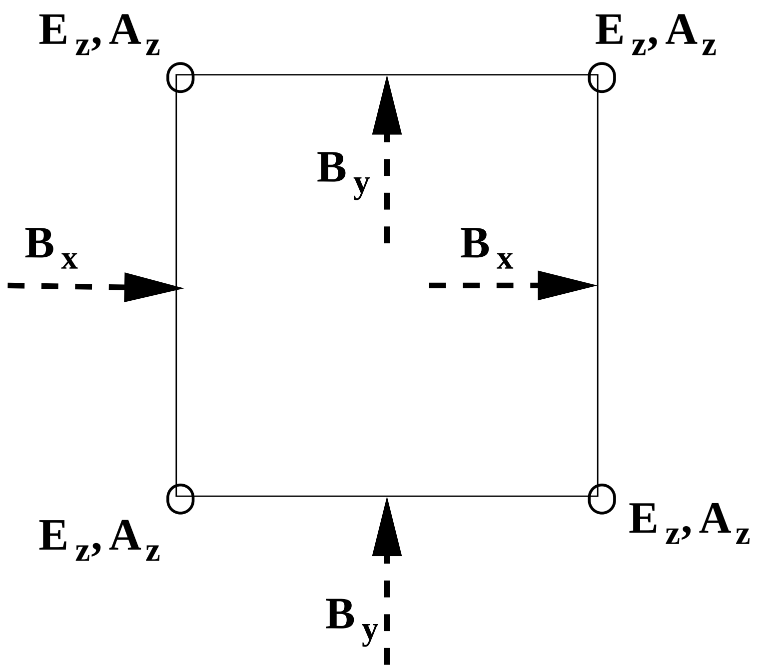
= time evolution of average normal component $\bar{B}_n$ of $\vec{B}$

$\Rightarrow \oint \vec{B} \cdot \vec{n} dS = 0$ on discrete level!!

because boundary of boundary vanishes (or contributions cancel)

CT on structured grids

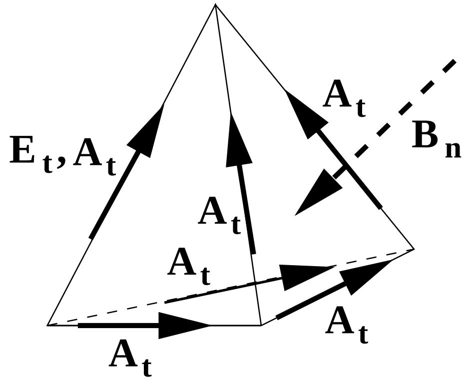
$$\frac{\partial \int_1^2 \vec{B} \cdot \vec{n} dl}{\partial t} = \frac{\partial \bar{B}_n}{\partial t} \Delta l = (\vec{v} \times \vec{B})_2 - (\vec{v} \times \vec{B})_1$$



B_x and B_y reconstruct $\vec{B}$ in nodes

= CT (Evans & Hawley 1988)

CT on unstructured grids

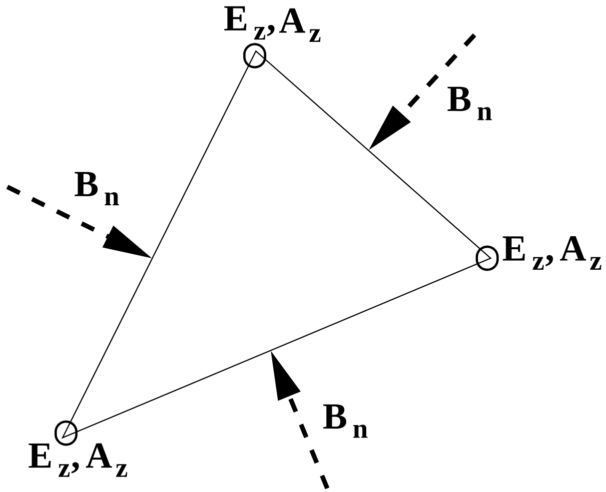


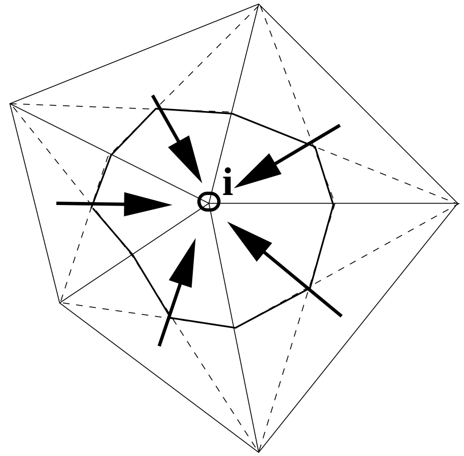
- represent $\vec{B}$ by $\bar{B}_n$: normal component on surfaces
- on unstructured grids, $\vec{B}$ can be reconstructed everywhere in the domain using vector basis functions (face elements for $\vec{B}$)
- update $\bar{B}_n$ using MU schemes (via MU interpolation of the reconstructed fields)
- this conserves the $\nabla \cdot \vec{B} = 0$ constraint at the discrete level up to machine accuracy
- this is tested for Faraday, Shallow Water MHD (system MUCT scheme)
- easy extensions: 2nd order (blended scheme), MHD, 3D, . . .

= generalization of CT to multi-dimensional methods on unstructured grids

Need vector basis functions: $\vec{P}_j$

(∼ face elements, from EM, e.g. Jin 93; Robinson & Bochev 2001 for MHD)





(1) reconstruct $\vec{B}$ in cell from $\bar{B}_n$ as

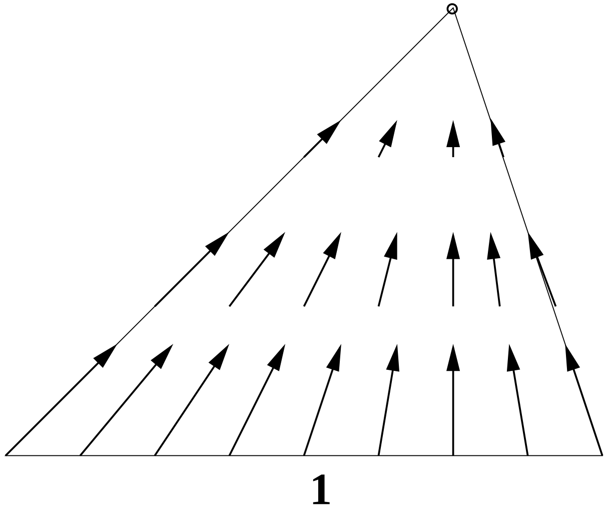
$$\vec{B}_{cell} = \sum_{j=1}^{3} \vec{P}_j \, B_{n,j}$$

(2) average $\vec{B}_{cell}$ to nodal $\vec{B}_i$ in upwind way

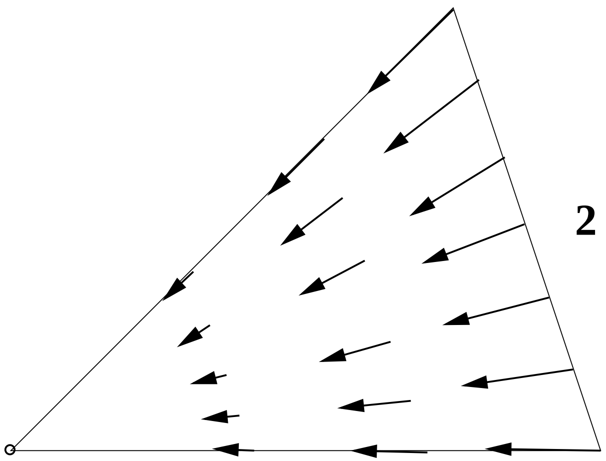
e.g. $\vec{P}_1$: normal component $\vec{P}_{1,n}$ constant on edge 1, vanishing on other edges

(2) Multi-Dimensional Upwind Constrained Transport

$$\vec{B}_{cell} = \sum_{j=1}^{3} \vec{P}_j \, B_{n,j}$$



$\vec{P}_1$



$\vec{P}_2$

e.g. $\vec{P}_1$: normal component $\vec{P}_{1,n}$ constant on edge 1, vanishing on other edges

(also higher order, quads, . . .: general concept)

(2) Multi-Dimensional Upwind Constrained Transport

$$\vec{B}_{cell} = \sum_{j=1}^{3} \vec{P}_j \, B_{n,j}$$

- $B_{n,j}$ such that $\nabla \cdot \vec{B} =$ constant $\equiv 0$ everywhere inside element
- B_n is continuous at element interfaces, so there also $\nabla \cdot \vec{B} = 0$

$\Rightarrow$ finite-element reconstructed solution satisfies $\nabla \cdot \vec{B} = 0$ everywhere!

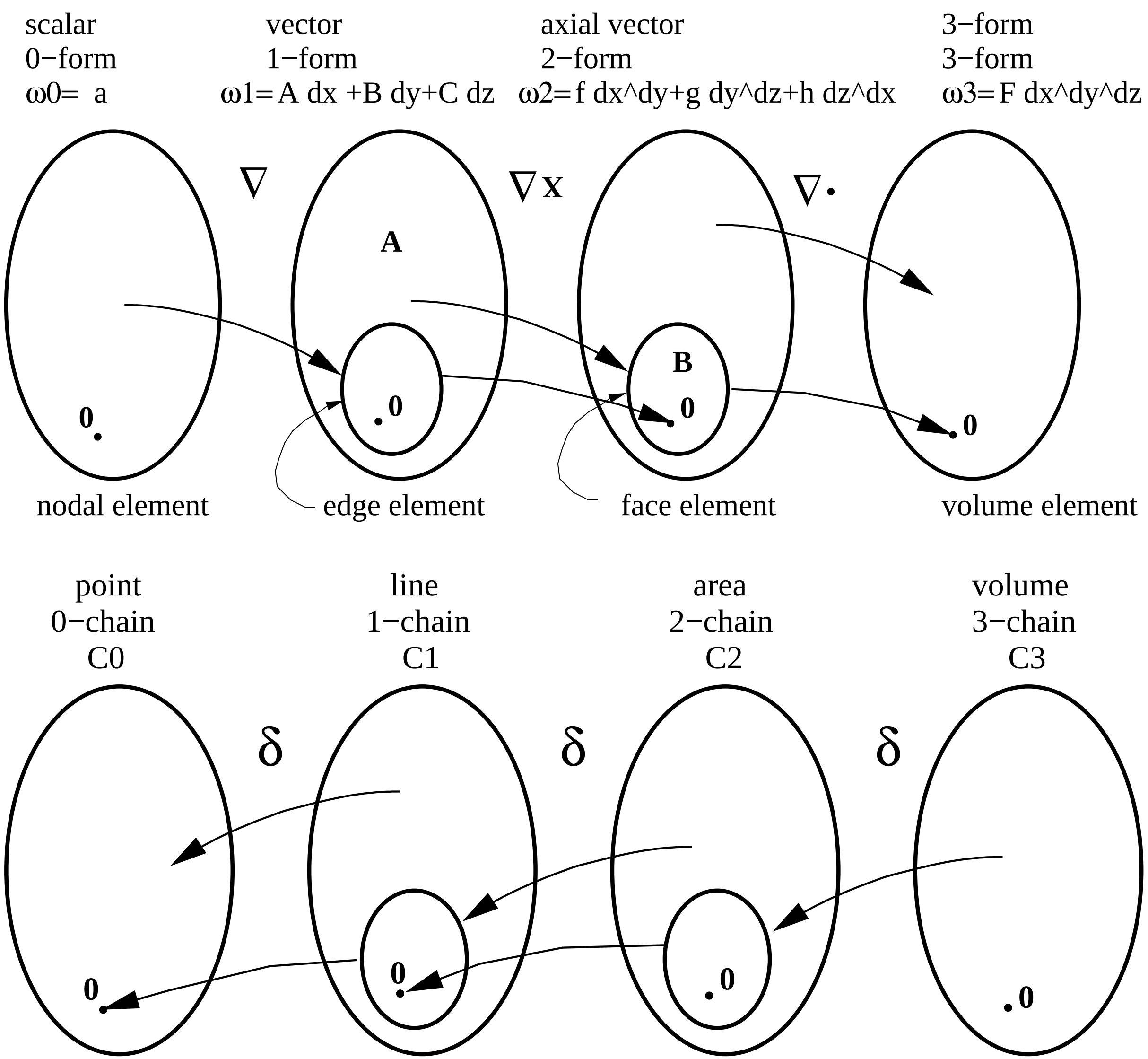
in triangle, for lowest order element:

$\vec{B}$ constant in space, B_n continuous

(on quad, or for higher order vector basis function:

$\vec{B}$ not constant in space, B_n continuous)

Interpretation: differential geometry



- physics = geometry
- numerics = geometry

$\Rightarrow$ in a consistent way!

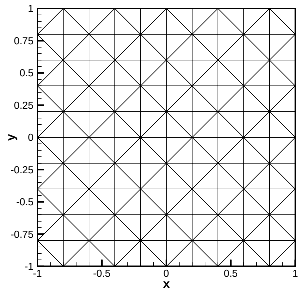
Application to 'Shallow Water' MHD system

(Gilman, ApJ 2000; De Sterck, Phys. Plasmas 2001)

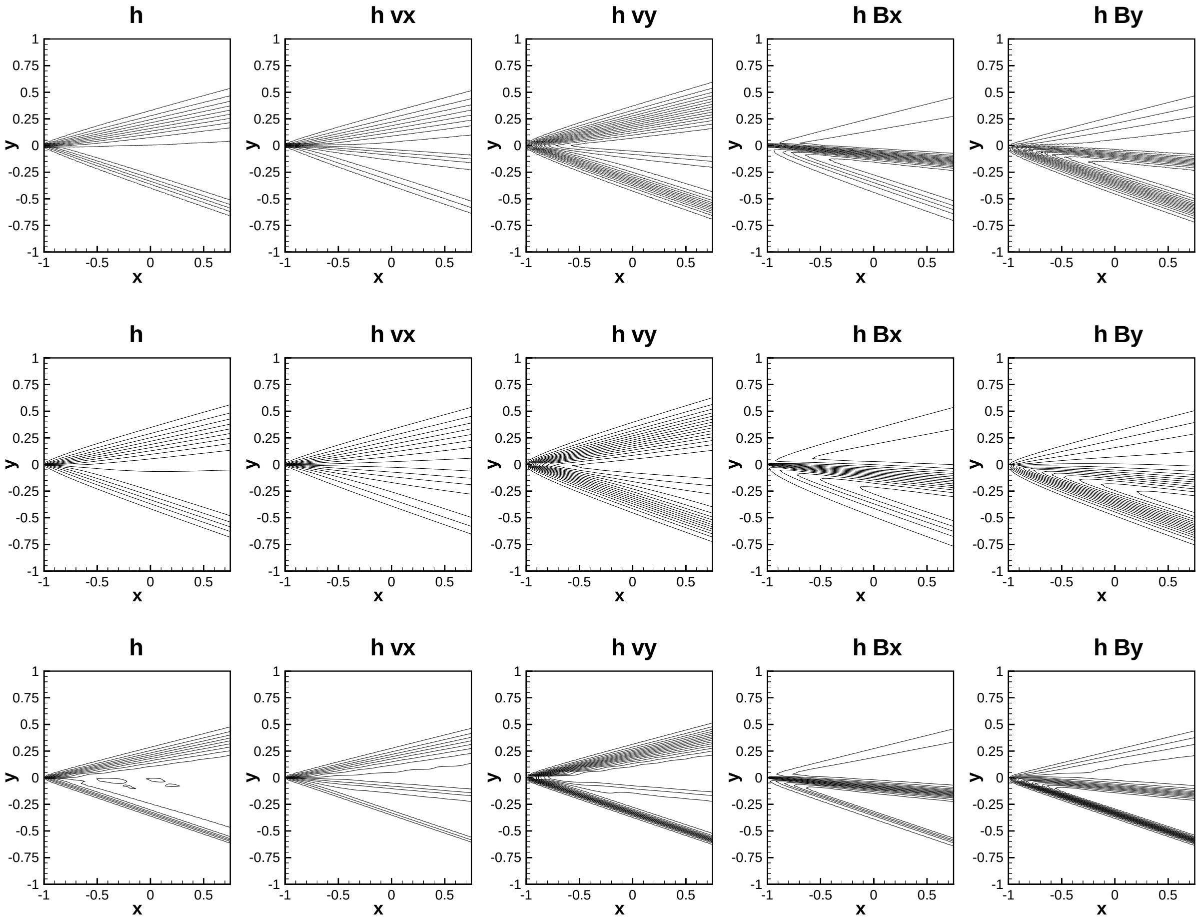
$$\frac{\partial h}{\partial t} + \nabla \cdot (h\,\vec{v}) = 0$$
$$\frac{\partial \vec{v}}{\partial t} + (\vec{v} \cdot \nabla)\vec{v} - (\vec{B} \cdot \nabla)\vec{B} + g\,\nabla h = 0$$
$$\frac{\partial \vec{B}}{\partial t} + (\vec{v} \cdot \nabla)\vec{B} - (\vec{B} \cdot \nabla)\vec{v} = 0$$

$$\nabla \cdot (h\,\vec{B}) = 0$$

- from MHD: incompressible, 2D variation, magnetohydrostatic equilibrium
- 4 wave modes: 2 magneto-gravity waves (nonlinear), 2 Alfvén waves (linear)
- one spurious 'div(B)'-wave (MHD!)



Steady Riemann problem

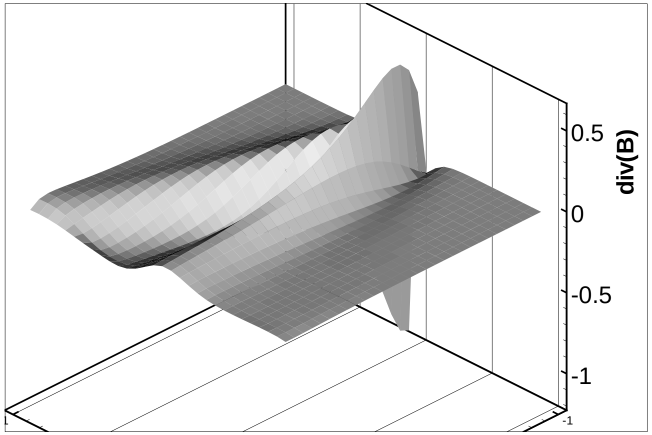


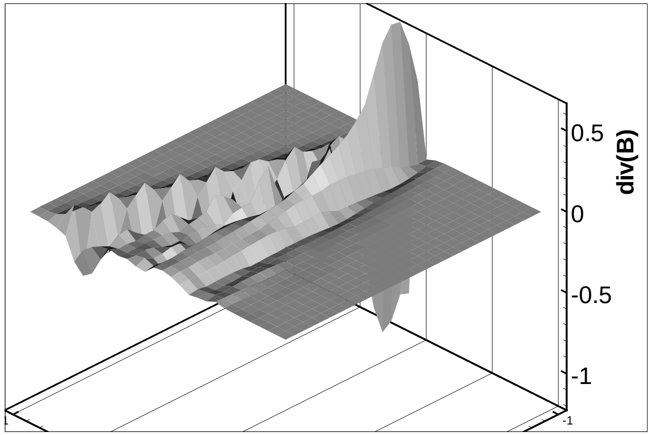
System N MUCT solution of the steady Riemann problem on a grid of 91×91 nodes. (No oscillations!)

First order Lax-Friedrichs finite volume solution of the steady Riemann problem on a grid of 90×90 finite volumes.

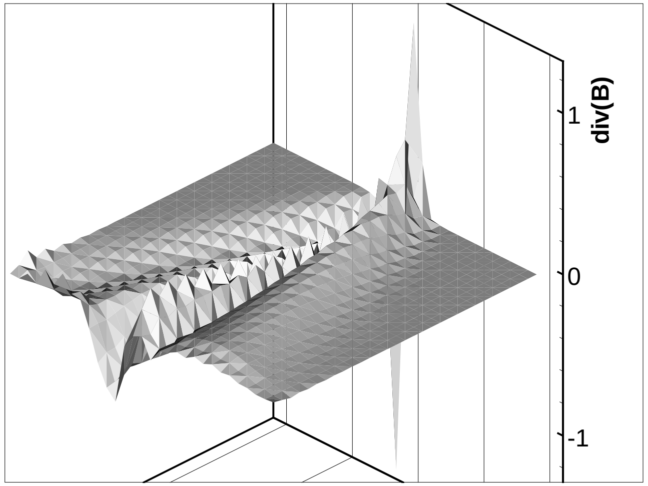
Second order Lax-Friedrichs finite volume solution of the steady Riemann problem on a grid of 90×90 finite volumes.

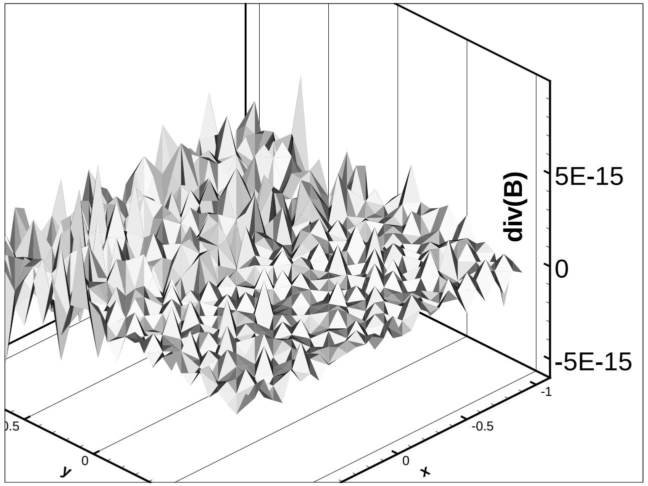
(2) Multi-Dimensional Upwind Constrained Transport





$\nabla \cdot \vec{B}$ for the first order (left) and second order (right) Lax-Friedrichs simulation of the steady Riemann problem on a grid of 30×30 finite volumes.





$\nabla \cdot \vec{B}$ for the full system N (left) and system N MUCT (right) simulation of the steady Riemann problem on a grid of 31×31 nodes.

(3) Java Taskspaces for Grid Computing

(with Rob Markel (master thesis project))

Computational Grids

- heterogeneous networks of geographically distributed computers

  example: SETI at home

  example: two large parallel computers connected through the internet

  example: linux clusters connected through very fast long-distance networks ("Teragrid")

- good for loosely coupled applications: task farming (small amount of communication, e.g. calculating π, statistics, computational biology)

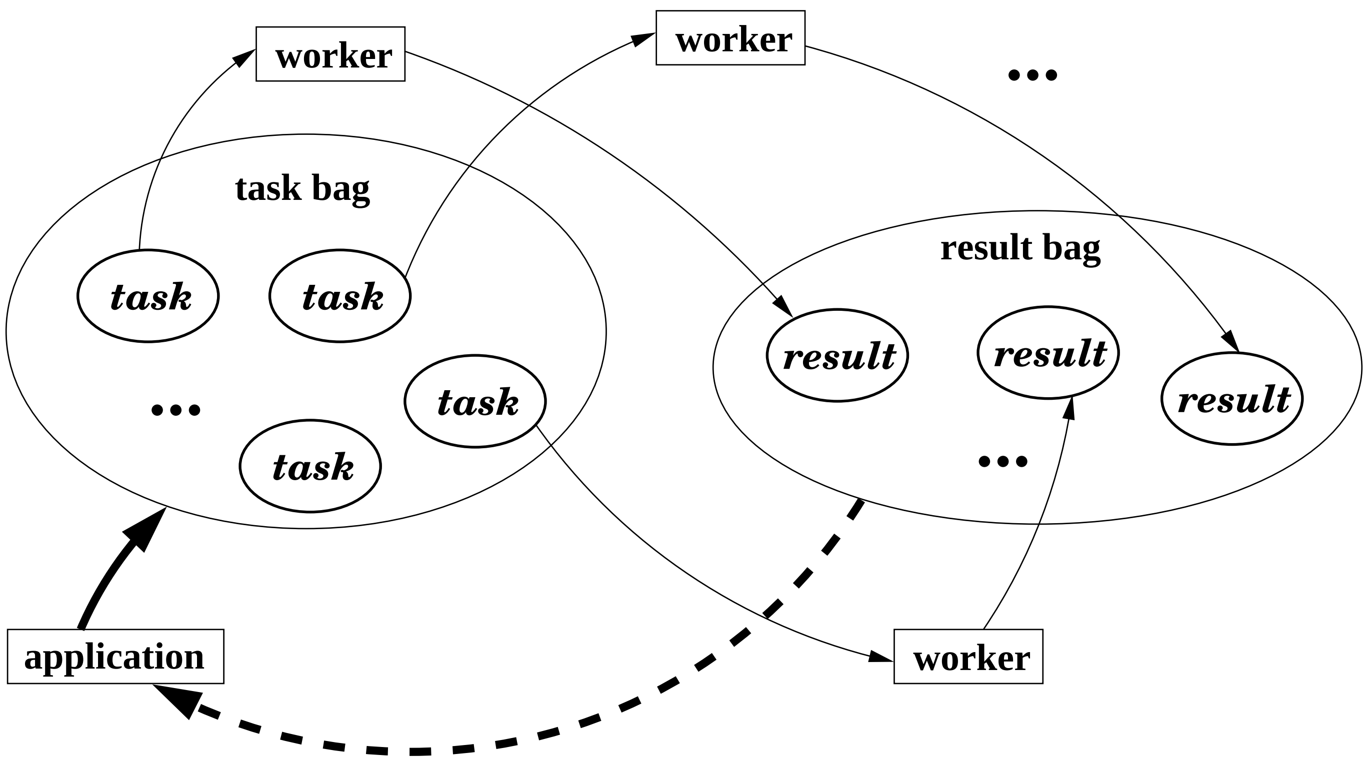
- more difficult for tightly coupled applications (network latency, but improving fast)

- "standard approach": GLOBUS and MPICH-G2

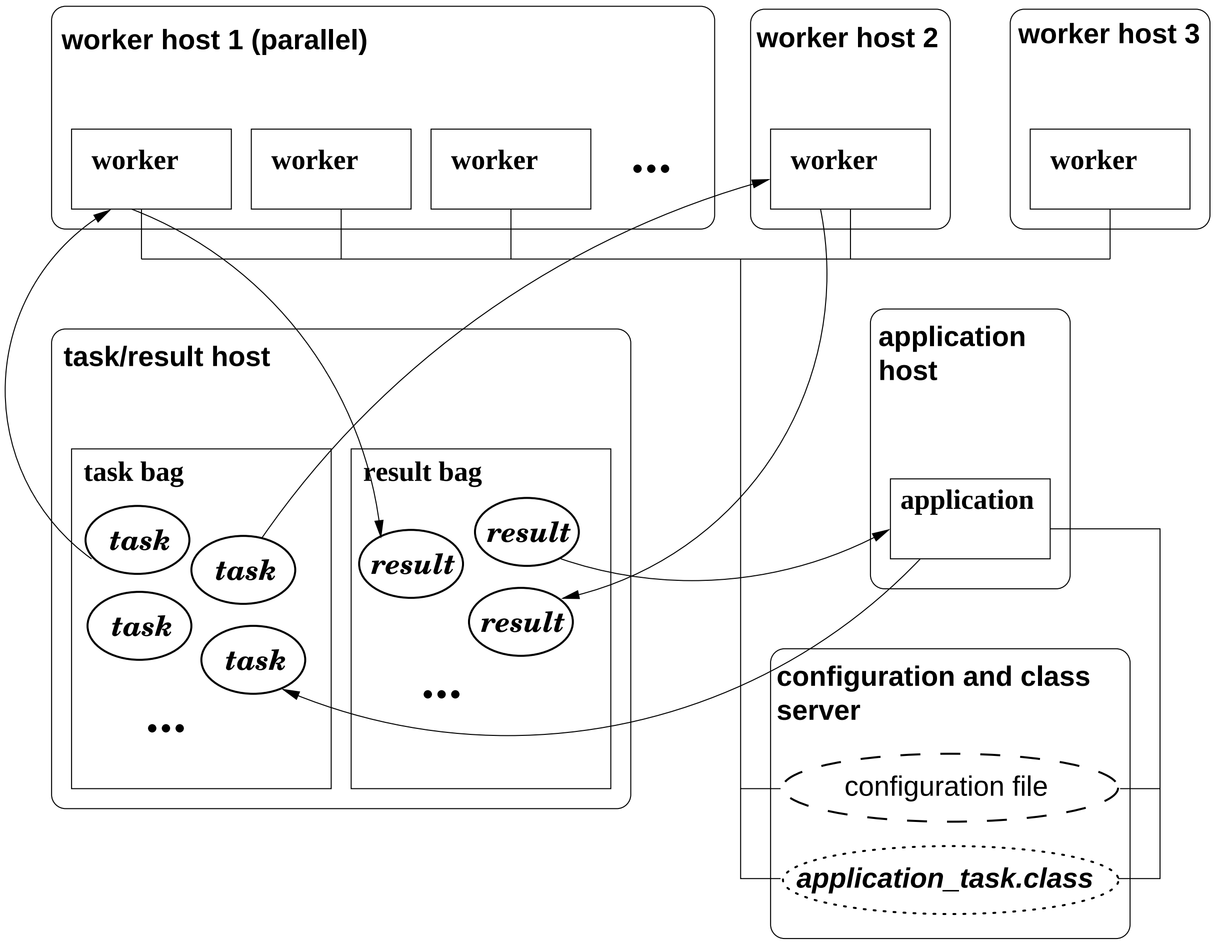
- our approach: Java Taskspaces

Distributed Computing Model Based on Tuple Space ideas

- a Tuple Space contains Task Objects
- decoupled in space and time
- self-configuring, no central coordination, flexible network topology
- natural loadbalancing, scalability, fault tolerance
- Tuple Spaces pioneered in the late 70s (Linda, JavaSpaces, TSpaces, ...)



Implementation in Java



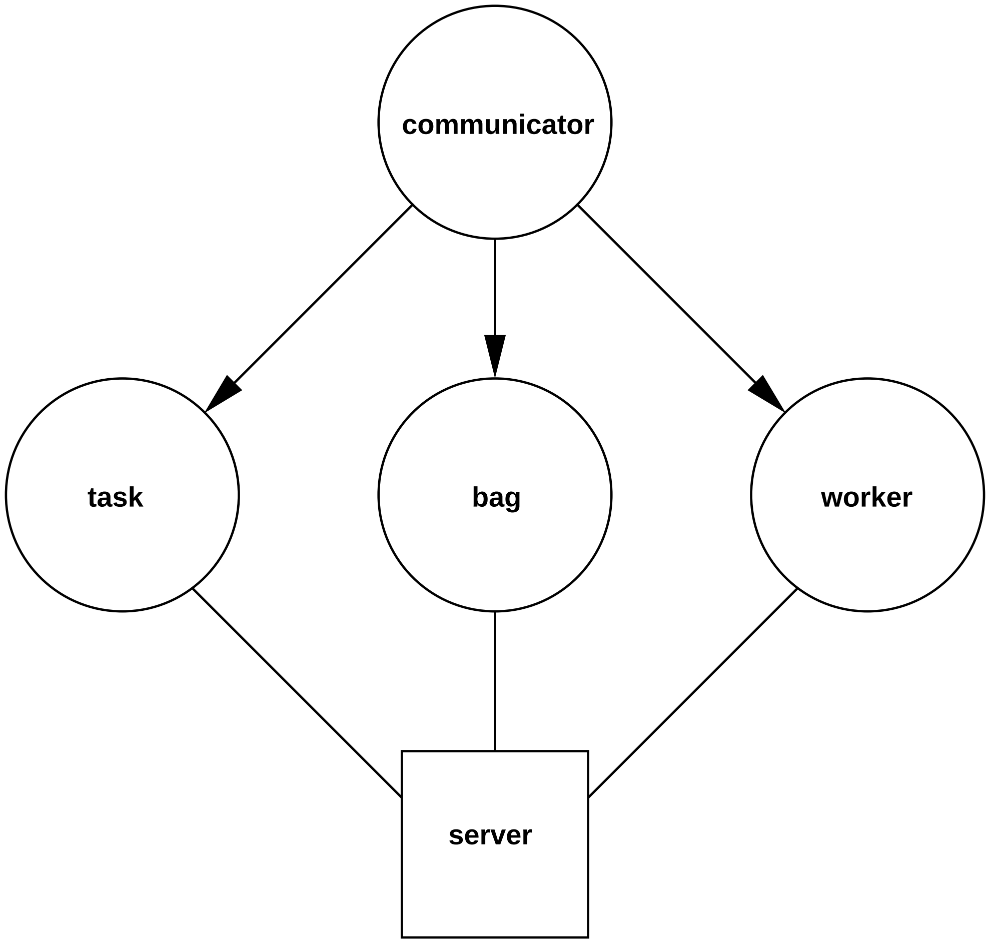
- platform independence
- object-oriented: Task Object = data + methods (code) , code downloaded
- “dumb” generic workers, easy to install, can run continuously
- very compact (Java is a complete, high-level language; workers $\sim$ 1kB)
- security: digitally signed .jar files are downloaded

Guidelines for design and implementation:

- simplicity
- compactness
- no legacy restrictions
- developed 'from scratch' (no Jini and JavaSpaces, TSpaces, Linda)
  - simple, compact, one layer
  - no overhead, streamlined for grid computing
  - full control (no dependence on software support that can be unreliable ...)
- high throughput computing
  - platform independence $\neq$ optimal performance on one parallel machine

    $\longrightarrow$ 'high performance computing' (MFlop ...) is not primary goal
  - efficienct use of general, commodity resources through reduced complexity, flexible network topology $\longrightarrow$ high throughput

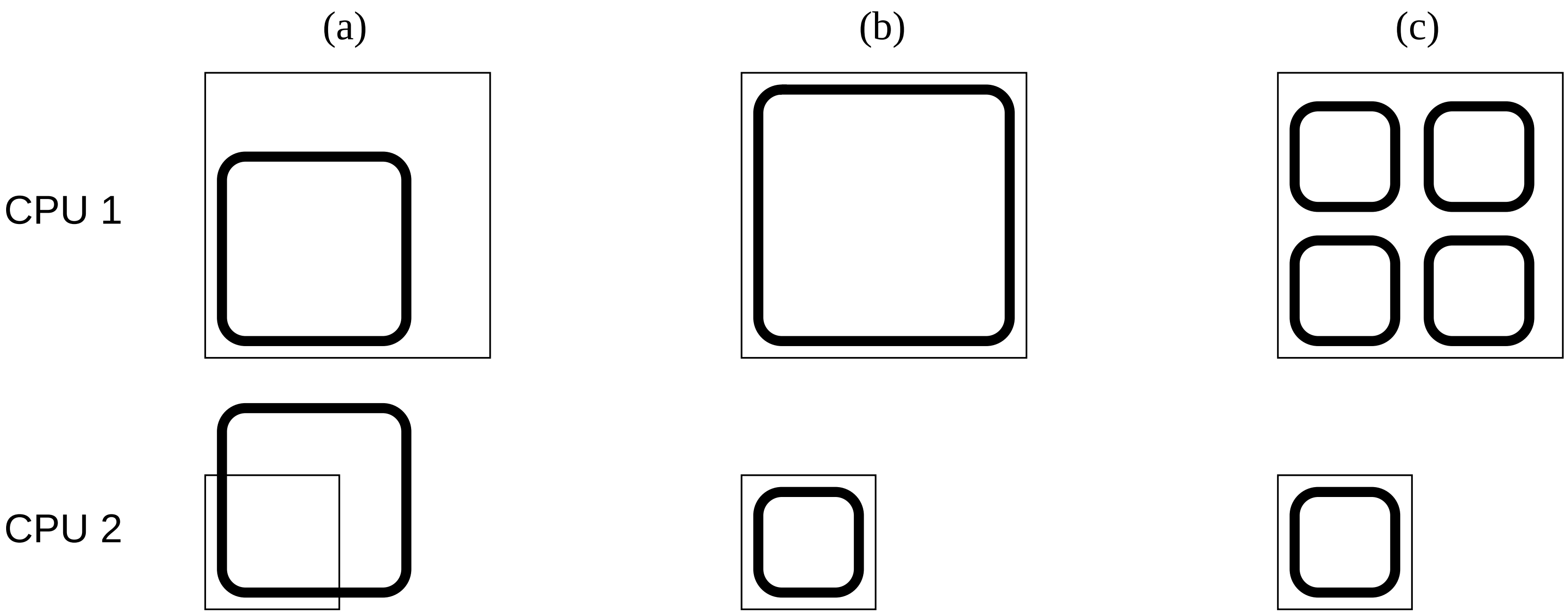
Java design and implementation:



- all classes extend 'Communicator'
  - read
  - write
- communication: send generic Objects over ObjectStreams associated with Sockets
- classes downloaded from http server using URLClassLoader

Some more potential conceptual advantages of Java Taskspaces:

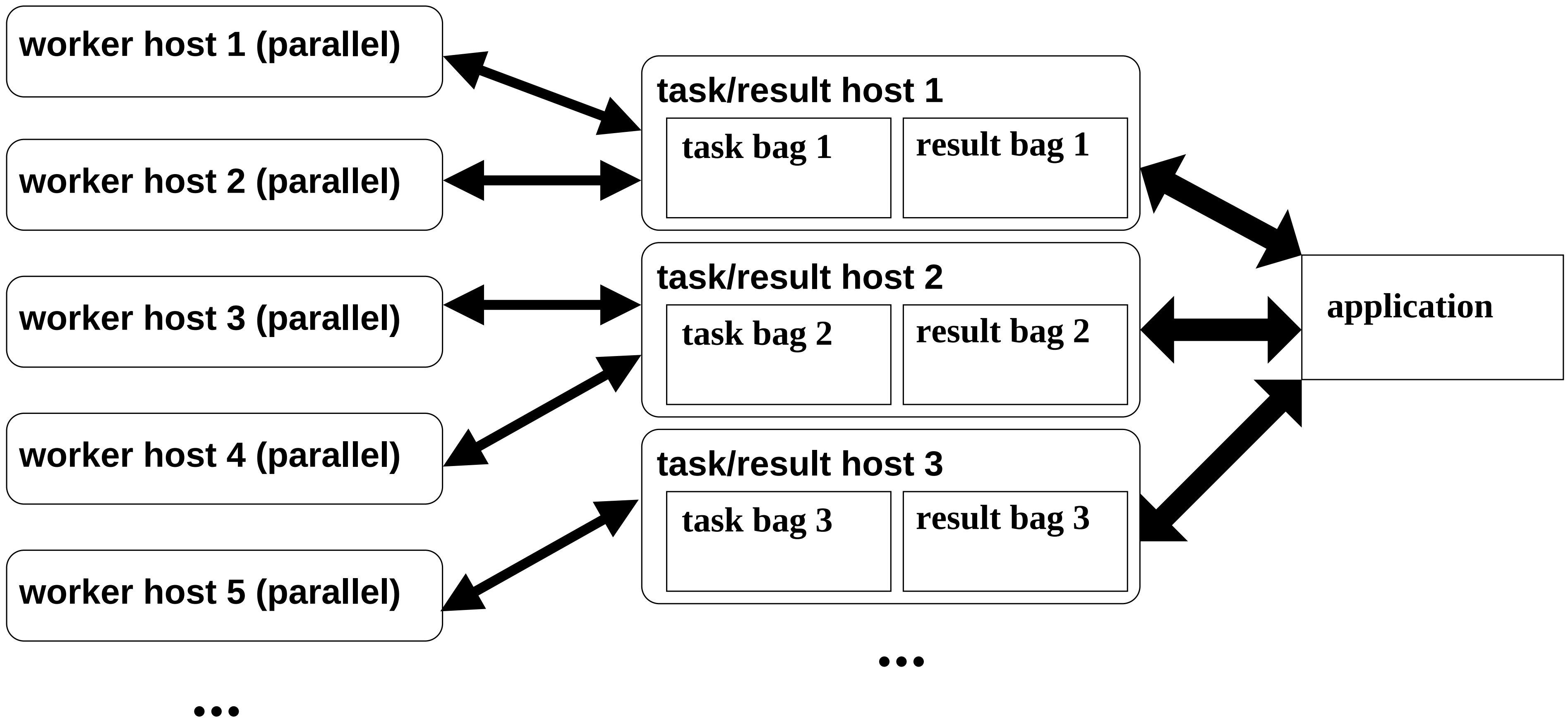
- natural automatic loadbalancing on heterogeneous grids



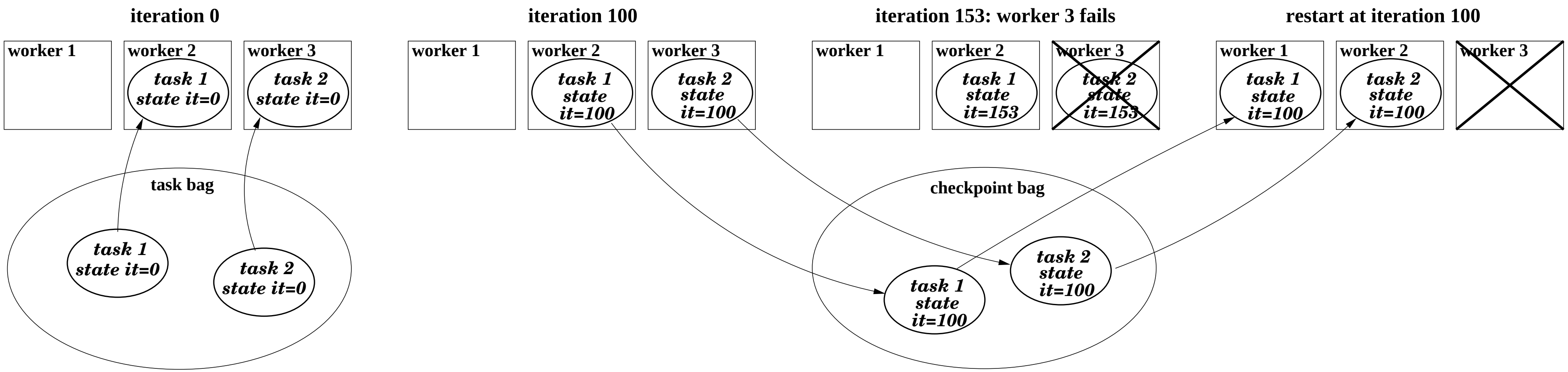
(b) custom task size: complex solution, depending on machines and application

(c) automatic loadbalancing: small granularity, multiple tasks on fast/large processors, simple solution

- natural scalability through multiple task/result bags



- natural fault tolerance
  - if no communication: resubmit tasks, or send multiple copies of tasks
  - if communication: natural checkpointing strategy by sending task objects containing state to checkpoint bags



- problems still under consideration:
  - registration, synchronization of workers
  - queue reservation, start remote workers (use GLOBUS ...)

Applications without communication ('task farming')

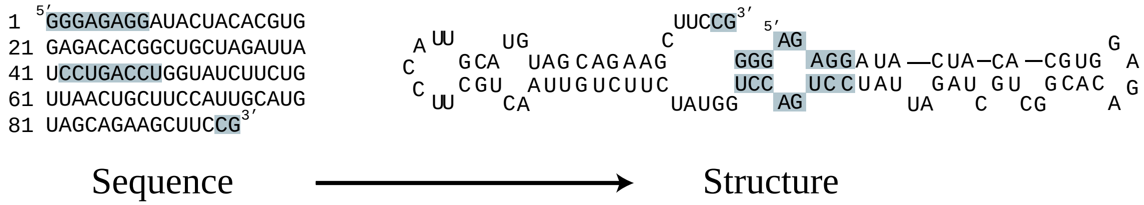
Calculating π

(1000 tasks)

Computer	Location	#Worker Processors	#Tasks Processed
Experiment 1			
BlueHorizon IBM SP	San Diego, CA (SDSC)	16 (Power3 375 MHz)	791
BabyBlue IBM SP	Boulder, CO (NCAR)	4 (Power3 375 MHz)	209
Experiment 2			
Newton Sun Server	Boulder, CO (CU)	1 (USparcIIi 360 MHz)	37
Laptop MS Windows	Boulder, CO (wireless)	1 (233 MHz P2)	29
BlueHorizon IBM SP	San Diego, CA (SDSC)	8 (Power3 375 MHz)	376
Grandprix Linux PC	Boulder, CO (CU)	1 (2.0 GHz P4)	370
BabyBlue IBM SP	Boulder, CO (NCAR)	4 (Power3 375 MHz)	188
Bagwan Linux PC	Boulder, CO (CU)	Task/application host	
Amath Sun server	Boulder, CO (CU)	Configuration server	

RNA, Universal Catalysis, and the Origin of Life: Virtual Experiments on Computational Grids

(with Rob Knight, CU Boulder Molecular Biology, NSF proposal pending)

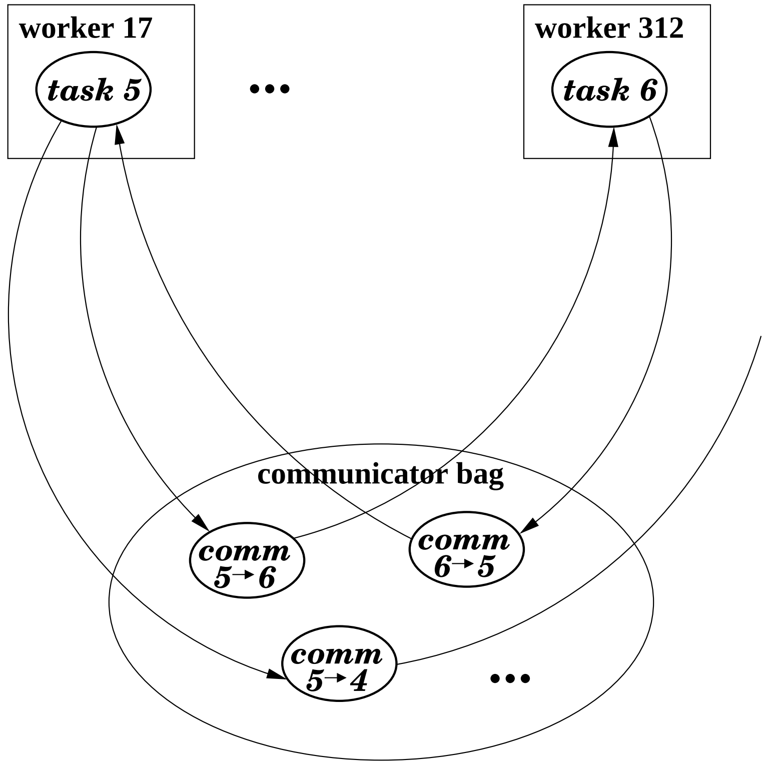


- **lab experiments**
  - **- random pools of RNA molecules (length $\sim$ 80) catalyze arbitrary molecular reactions**
  - **- this universal catalysis may have started a primitive metabolism in an early "RNA world"**
  - **-** $10^{13} - 10^{15}$ **random molecules are certainly sufficient**
- **virtual experiments on computational grids**
  - **- estimate how many random molecules could have been sufficient for an RNA world**
  - **- algorithm: every tasks (1) generates random RNA sequences, (2) computes folding structure, (3) compares with database of known catalytic sequences and structures**
  - **- no communication, task farming**
- **use existing C code by wrapping statically linked C executable in .jar file and by downloading appropriate executable based on system information (operating system)**

Applications with communication

- Jacobi, Conjugate Gradient iterative methods
- MHD simulations ...

approach 1: communication bag

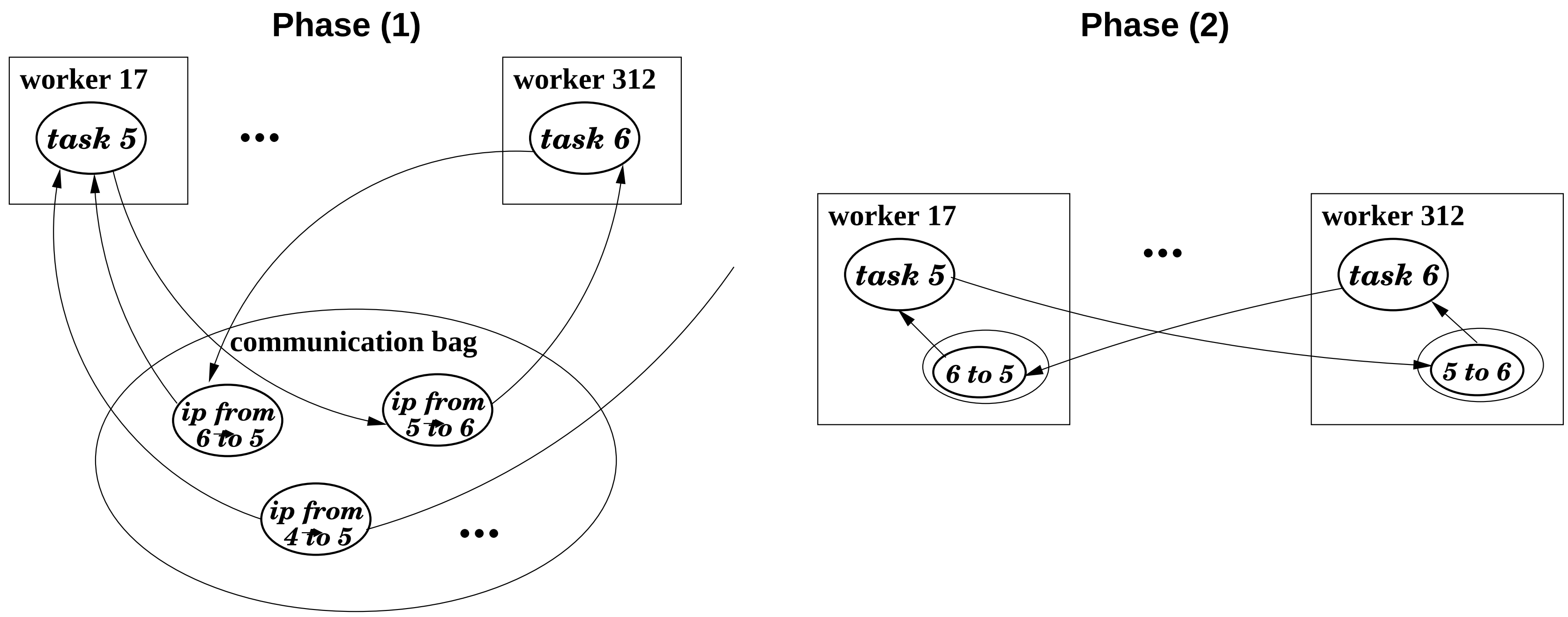


- faithful to tuple space concept

- communication bottleneck

approach 2: set up direct communication

- every worker has its own communication bag
- two phases:
  - Phase (1): set up communication pattern
  - Phase (2): direct communication
- retains flexibility in topology, configurability, fault tolerance through checkpointing



global communication

- through 'intelligent bag' (add, substract, max, min, ...)
- possibly hierarchically, like MPI

work in progress

- Jacobi with message bag and with direct messages
- ready to do scaling tests on
  - 4 IBM SP: in San Diego (SDSC, > 1000 processors), Boulder (NCAR, > 700 processors and 64 processors), UMichigan (48 processors)
  - several SGI Origin2000: in Illinois (NCSA, > 500 processors)
  - large Intel linux cluster: in New Mexico (512 processors)
  - workstations in Boulder, Erlangen
- main issue: queueing systems, need reservation (GLOBUS ...)

$\Rightarrow$ full functionality of Cactus+Globus+MPICH-G2!! (SC2001 Bell award)

+ many additional advantages

- Possible Applications for Grid Computing in Space Physics