A Gentle Introduction to High Performance Computing

V. Sundararajan
Scientific and Engineering Computing Group
Centre for Development of Advanced Computing
Pune 411 007
vsundar@cdac.in

Plan

- Why HPC?
- Parallel Architectures
- Issues in using parallel machines?
- Application examples

Definition

Supercomputing

Parallel computing: Parallel Supercomputers

High Performance Computing: includes computers, networks, algorithms and environments to make such systems usable. (range from small cluster of PCs to fastest supercomputers)

CPU Speed

Physical Limits of single processor speed– 2007 (Predicted by Paul Messina in 1997)

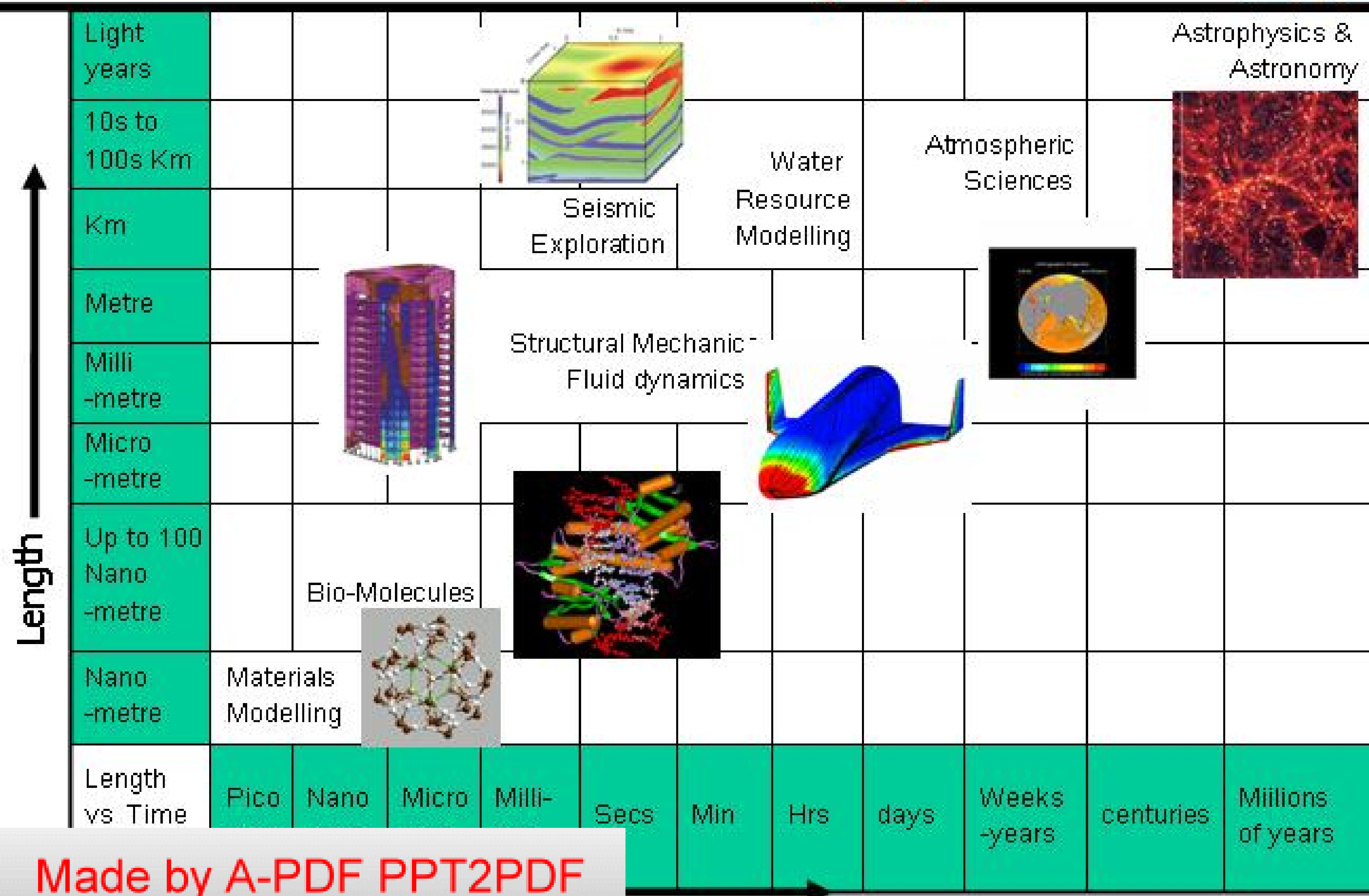
Pentium	3600 MHz	<0.3 ns
Light travels	30 cm in	1 ns

Signal is already 10% of its speed

Recently, Increased cores at lower clock speed added value

Red-Shift: Reduction in clock speed

Killer or Grand Challenge Applications



If we focus on some set of priorities, HPC can be guaranteed to produce important results. The grand challenge applications should not be regarded as problems that can be "solved". Instead they should be regarded as focus areas in which improved products, new products, and new engineering or scientific insights can be generated. Additionally, we can guarantee that eventually fallout will occur in other computational science and engineering areas.

David J. Kuck

HPC: Challenges for Future Systems

Why HPC?

To simulate a bio-molecule of 10000 atoms
Non-bond energy term ~ 10^8 operations
For 1 microsecond simulation ~ 10^9 steps
~ 10^{17} operations
On a 500 MFLOPS machine (5×10^8 operations per second) takes 2×10^9 secs (About 60 years)
(This may be on a machine of 5000 MFLOPS peak)

Need to do large no of simulations for even larger molecules

Parallel Architectures

1. Single Instruction Single Data (SISD) Sequential machines.
2. Multiple Instructions Single Data (MISD)
3. Single Instruction Multiple Data (SIMD)

   Array of processors with single control unit.

   Connection Machine (CM - 5)

4. Multiple Instructions Multiple Data (MIMD)

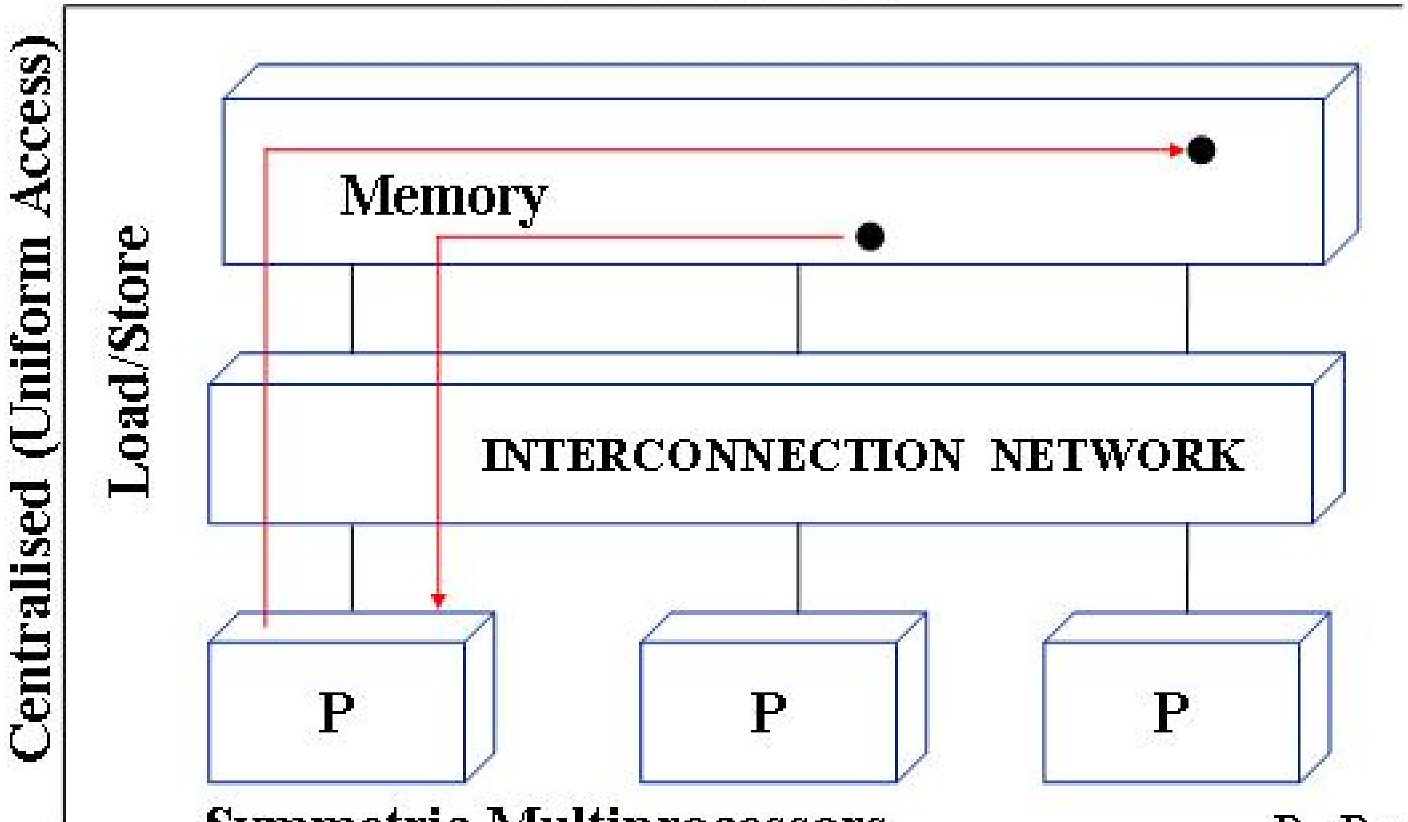
   Several Processors with several Instructions and Data Stream.

   All the recent parallel machines

SMPs, Contellations, Clusters, Massively Parallel Processors MPP

Tightly Coupled MIMD

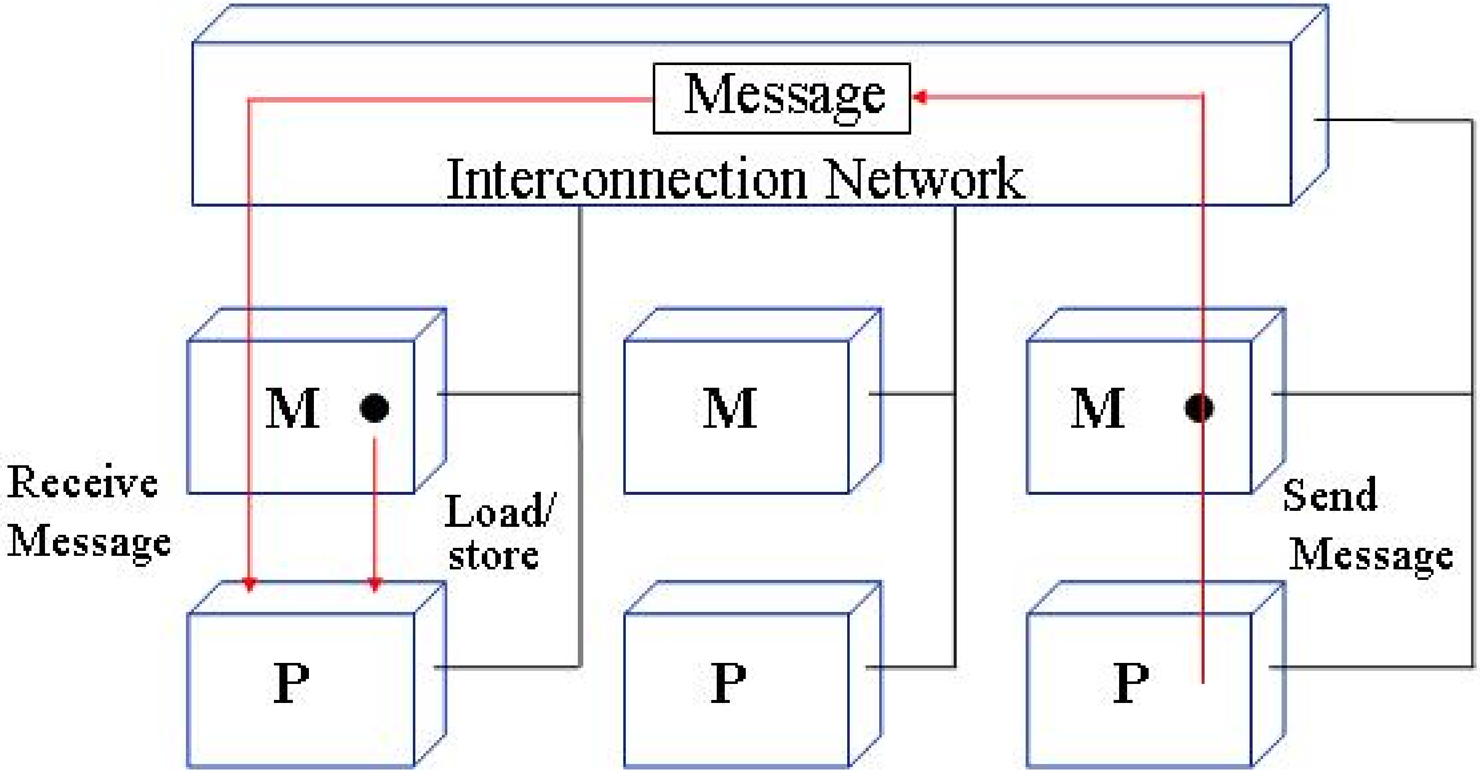
Shared Memory



Symmetric Multiprocessors

P - Processor

Loosely Coupled MIMD



Distributed Message-Passing Machines

Also Called Message Passing Architecture

M - Memory
P - Processor

Architectures

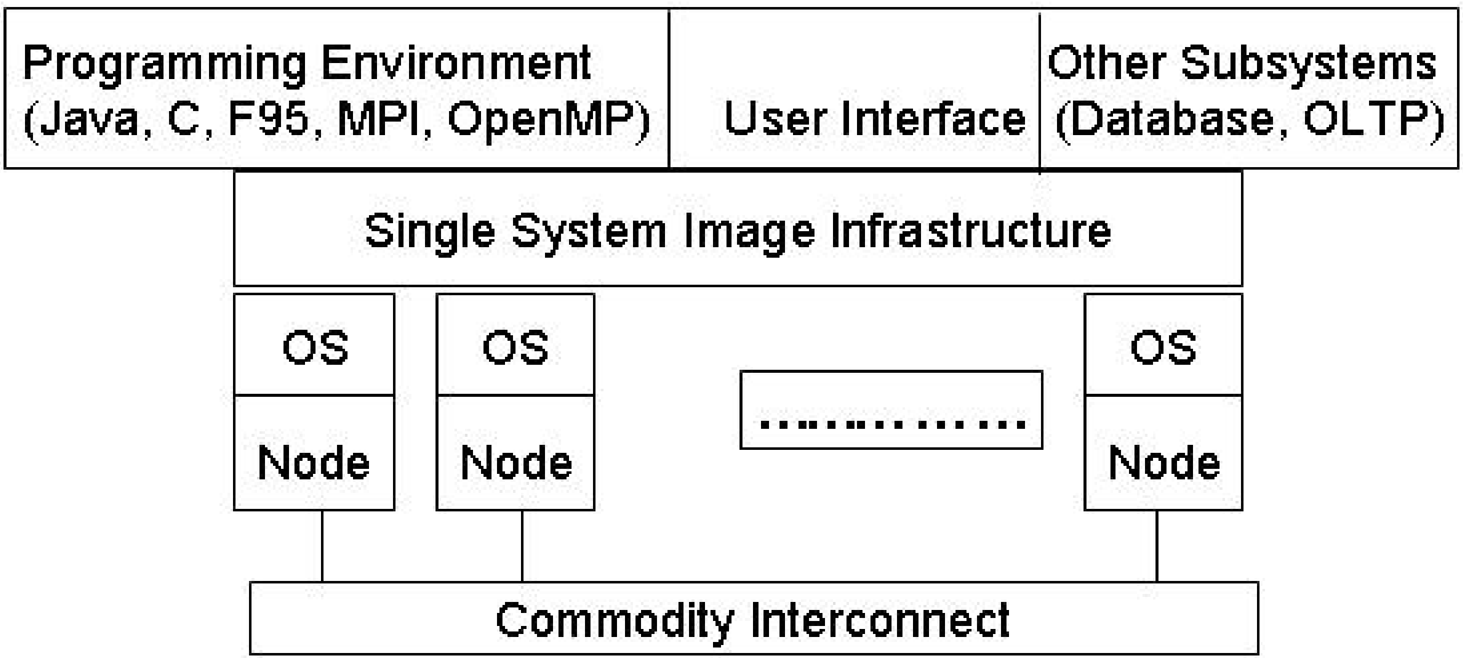
Shared Memory :

- Scalable up to about 64 or 128 processors but costly
- Memory contention problem
- Synchronisation problem
- Easy to code and make mistakes

Distributed Memory :

- Scalable up to larger no. of processors
- Message passing overheads
- Latency hiding
- Difficult to code

Cluster of Computers



Simulation Techniques

N-body simulations
Finite difference
Finite Element
Pattern evolution

→

Classic methods -
Do Large scale problems
Faster
New class of Solutions

Parallel Problem Solving Methodologies

- Data parallelism
- Pipelining
- Functional parallelism

Parallel Programming Models

1. Directives to Compiler (OpenMP, threads)
2. Using Parallel libraries (SCALAPACK, PARPACK)
3. Message passing (MPI)

Programming Style

SPMD : Single program multiple Data.

Each processor executes the same program but with different data.

MPMD : Multiple programs, multiple data.

Each processor executes a different program. Usually this is a master slave model.

Performance Issues

1. **Speed up** $= \dfrac{\text{Time for sequential code}}{\text{Time for parallel code}}$

$$S_p = \frac{T_s}{T_s} \qquad\qquad 1 \leq S_p \leq P$$

2. **Efficiency** $E_p = \dfrac{S_p}{P}$ $\qquad 0 < E_p < 1$

$$E_p = 1 \Rightarrow S_p = P$$ 100% efficient

Communication Overheads

Latency

Startup time for each message transaction 1 μs

Bandwidth

The rate at which the messages are transmitted across the nodes / processors 10 Gbits / Sec.

Strongest argument

Amdahl's Law

$$S = \frac{1}{f + (1-f)/P}$$

f = Sequential part of the code.

Ex. $f = 0.1$ assume $P = 10$ processes

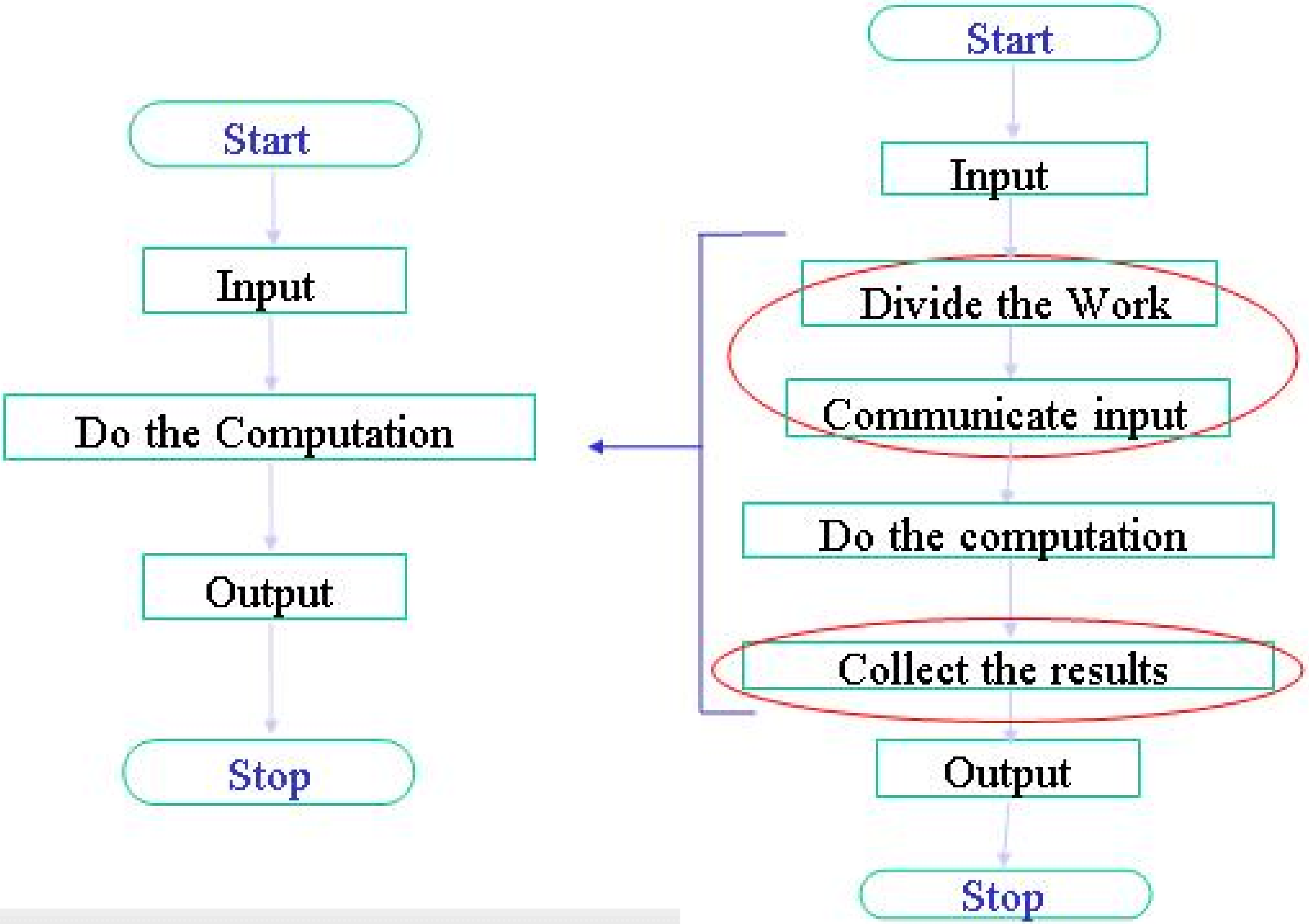
$$S = \frac{1}{0.1 + (0.9)/10}$$

$$= \frac{1}{0.1 + (0.09)} \cong 5$$

As $P \longrightarrow \infty$ $S \longrightarrow 10$

[illegible] maximum speedup possible.

Sequential vs Parallel

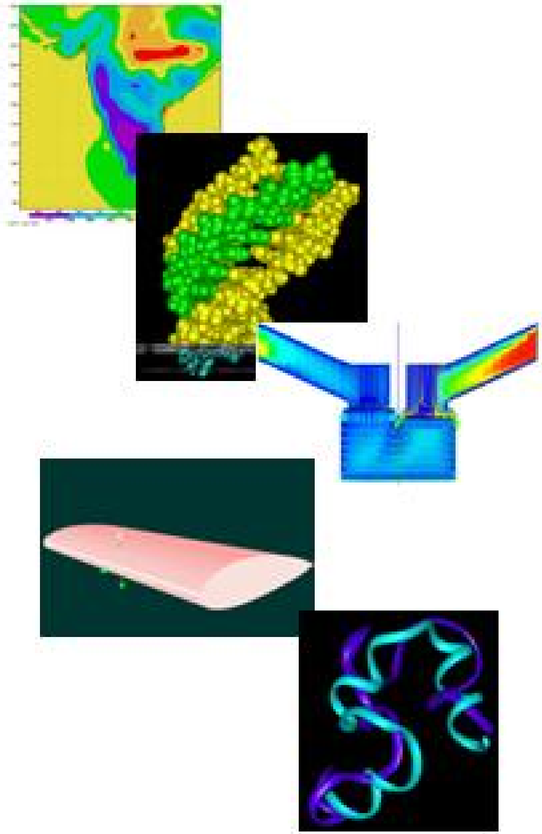


Sequential vs Parallel

Sequential	Parallel
Design algorithm step by step and write the code assuming single process to be executed	Redesign the same algorithm with the distribution of work assuming many processes to be executed simultaneously
f90 –o sum.exe sum.f gcc –o sum.exe sum.c Only one executable created	mpif90 –o sum.exe sum.f mpicc –o sum.exe sum.f Only one executable created
sum.exe Executes a single process	mpirun –np 8 sum.exe Executes 8 processes simultaneously
Single point of failure Good tools available for debugging	Cooperative operations but, multiple points of failure; Tools are still evolving towards debugging parallel executions.

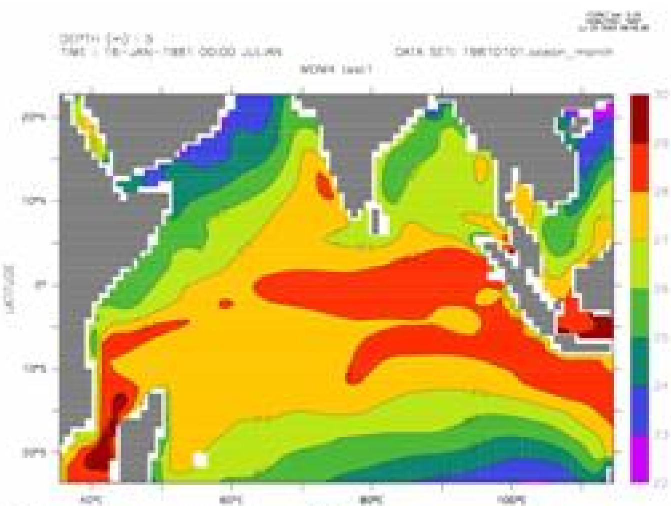
Scientific and Engineering Computing in C-DAC

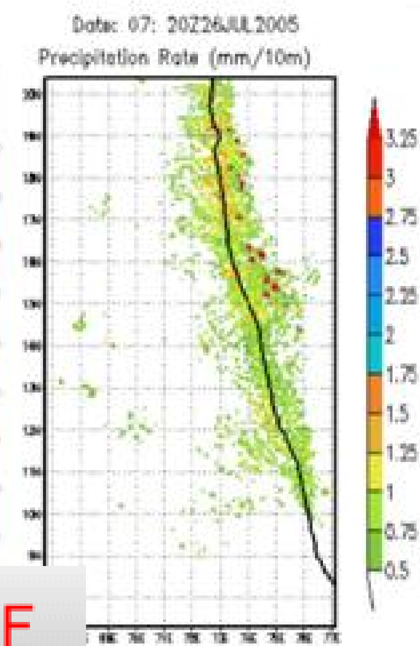
- *Computational Atmospheric Sciences*
- Bioinformatics
- *Seismic Data Processing*
- *Applied Evolutionary Computing*
- Computational Fluid Dynamics
- Computational Structural Mechanics
- Computational Materials Modelling

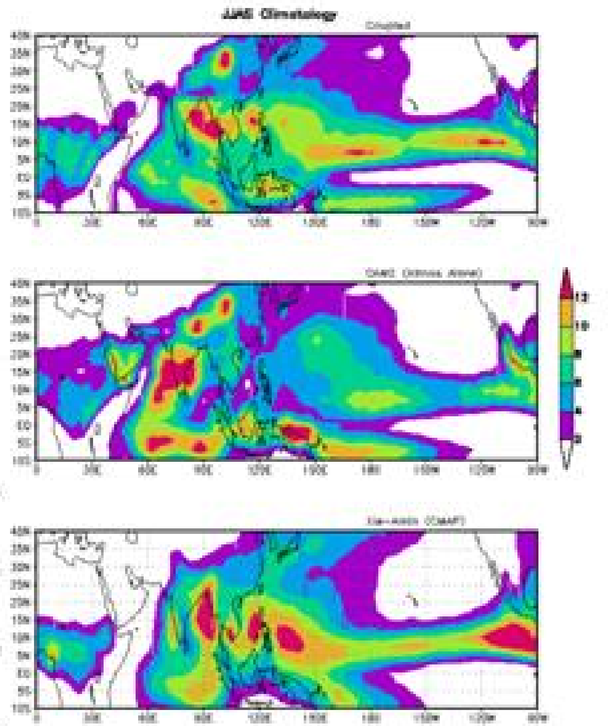


Computational Atmospheric Sciences

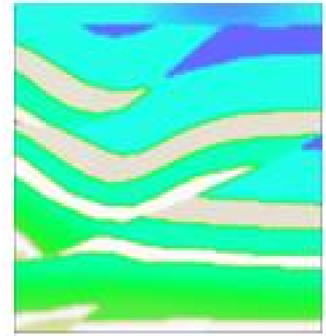
Ocean Modelling
Atmospheric Modelling
Coupled Ocean Atmosphere Models
Predicting Pollution by simulations







Seismic Data Processing

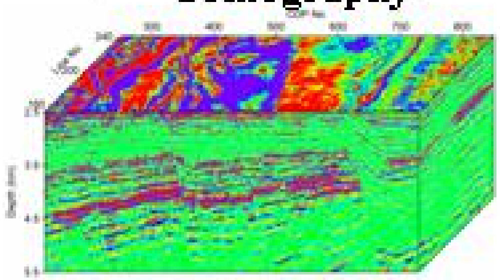


Seismic Modelling

Use: Exploration of Oil and Gas

Compute Intensive Algorithms:

- Modelling
- Migration
- Waveform Inversion
- Tomography



Seismic Migration

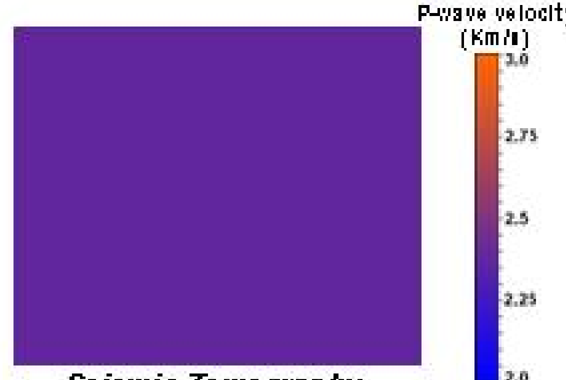
3D Onepass poststack depth migration

Run time on 64 CPUs of PARAM 10000 477mins

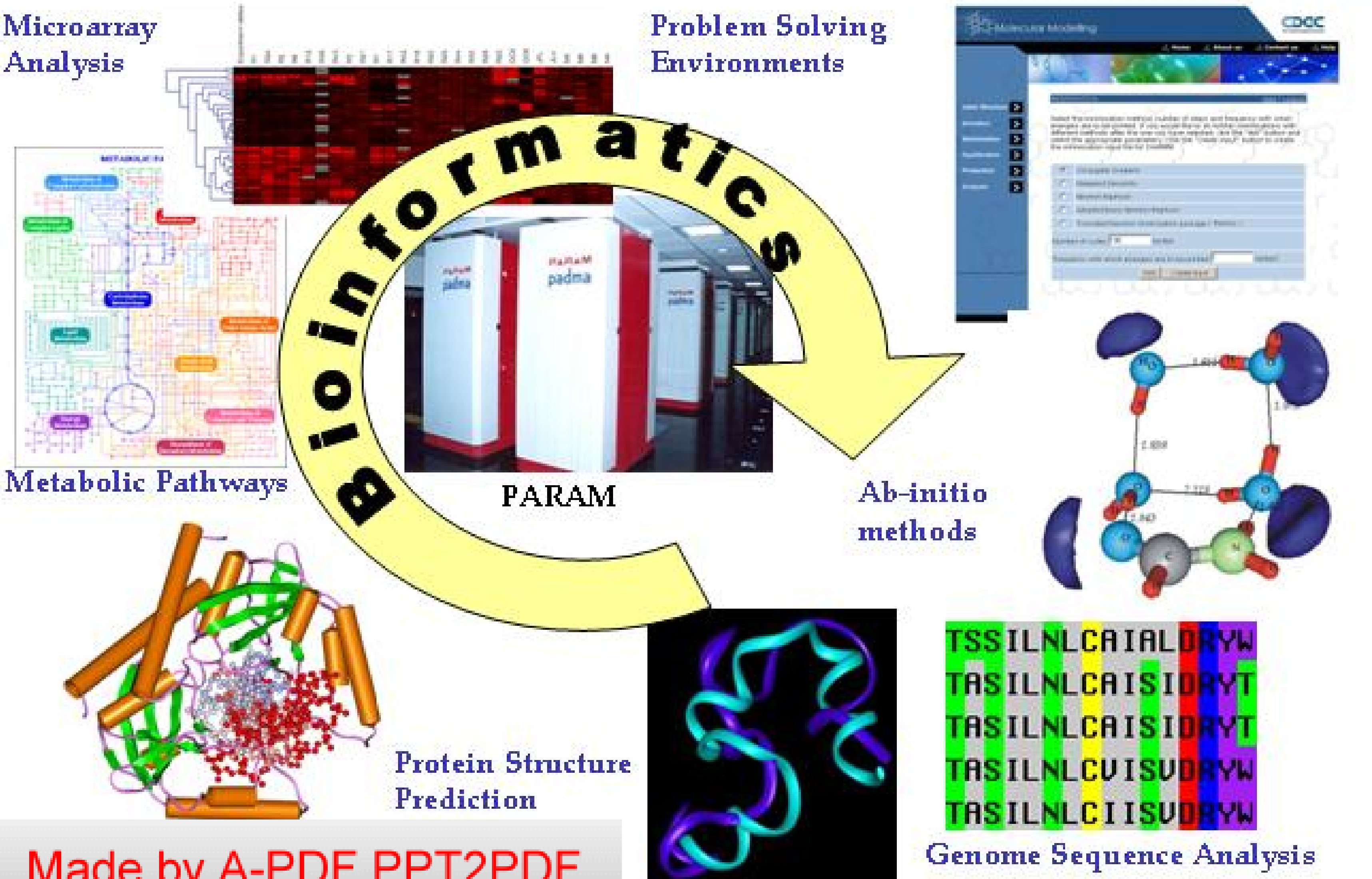
Run time on 16 CPUs of PARAM Padma 78 mins

2D Seismic Tomography using genetic algorithm

The reconstruction of the subsurface velocity model by seismic treveltime



Seismic Tomography


Microarray Analysis
Problem Solving Environments
Bioinformatics
Metabolic Pathways
PARAM
padma
Ab-initio methods
Protein Structure Prediction
TSSILNLCAIALDRYW
TASILNLCAISIDRYT
TASILNLCAISIDRYT
TASILNLCVISVDRYW
TASILNLCIISVDRYW
Genome Sequence Analysis

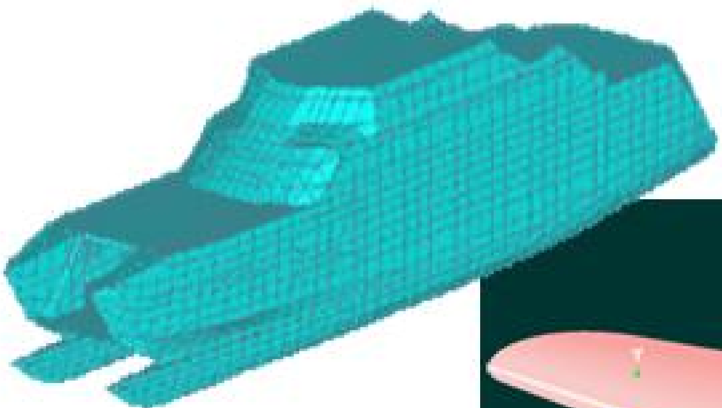
Computational Structural Mechanics

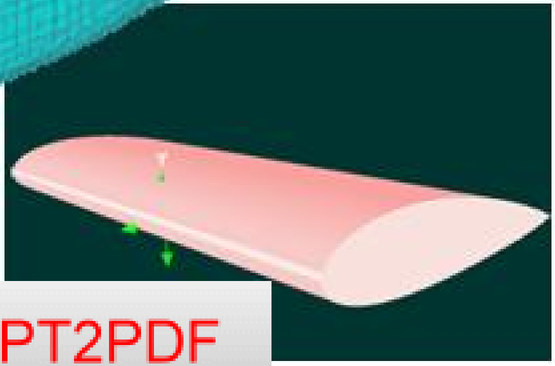
Structural Safety : 3-D Fracture Analysis on PARAM

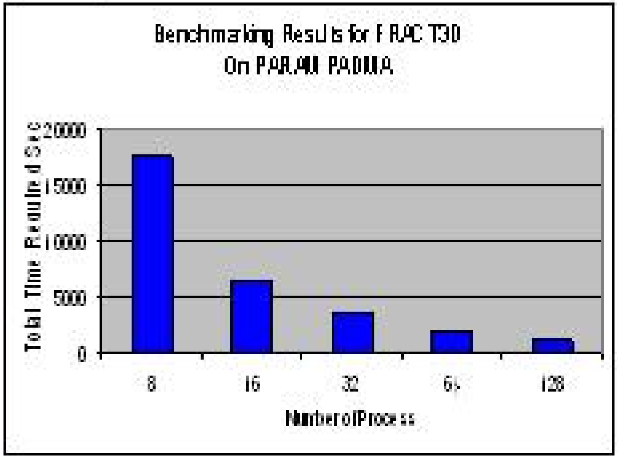
High Technology Material Modelling : FRP Composites Analysis

Smart Structures : Aerospace and Construction Industry

Earthquake Engineering : CAD Software Development





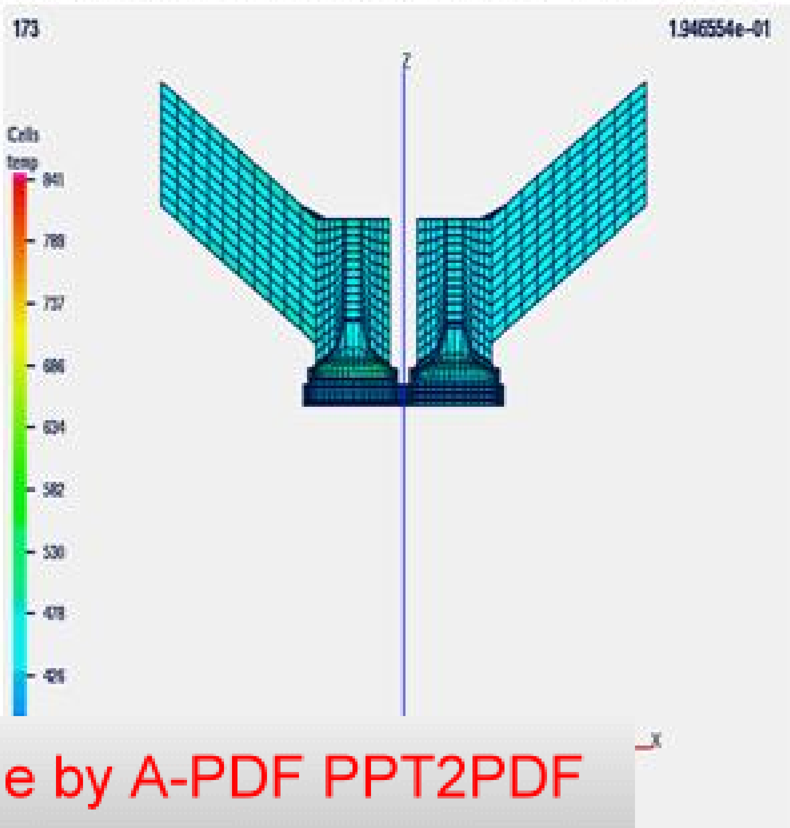


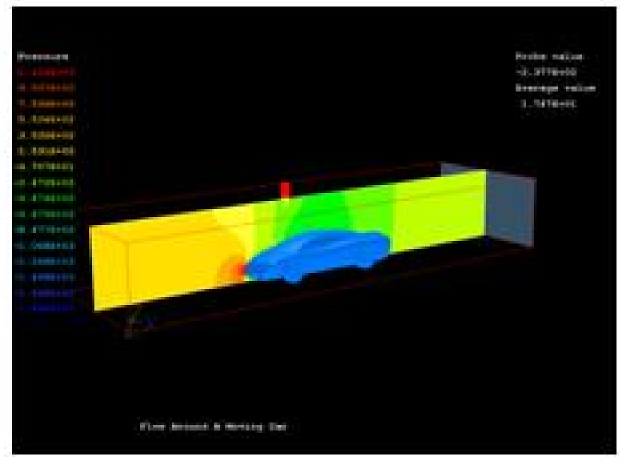
Computational Fluid Dynamics

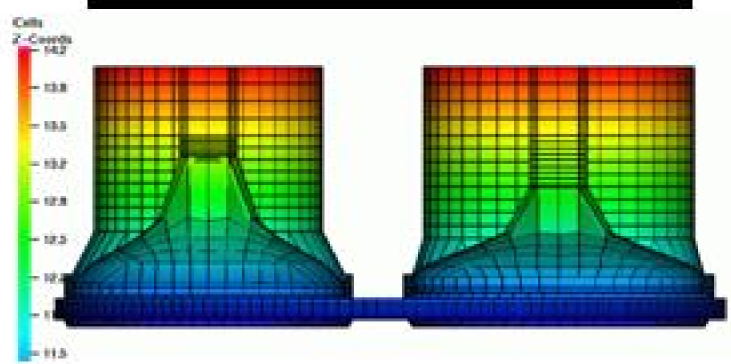
IC Engine Simulation

Flow Simulation over Re-launch vehicles (VSSC & ICAD Moscow)

Applications in Aerospace (with IIT Mumbai)

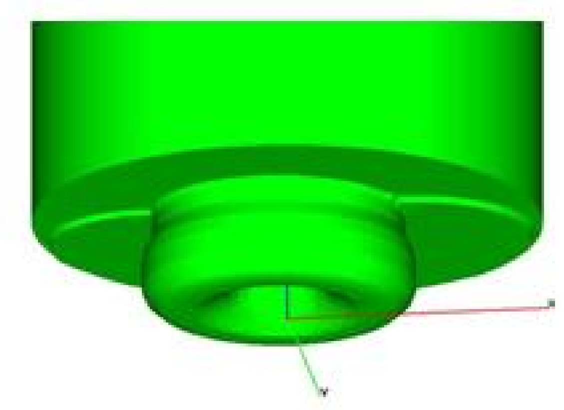


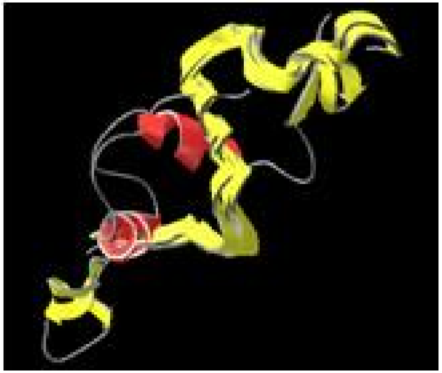




Applied Evolutionary Computing

- Evolving Solutions for
  - IC Engine design optimisation
  - Protein Structure
  - Similarity Search
  - Coal Mill modeling
- Monte Carlo Simulation of Faults



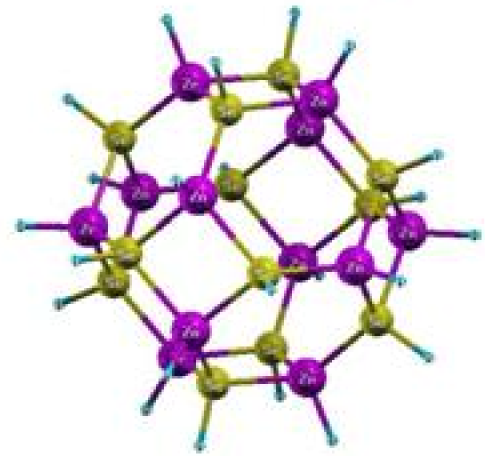


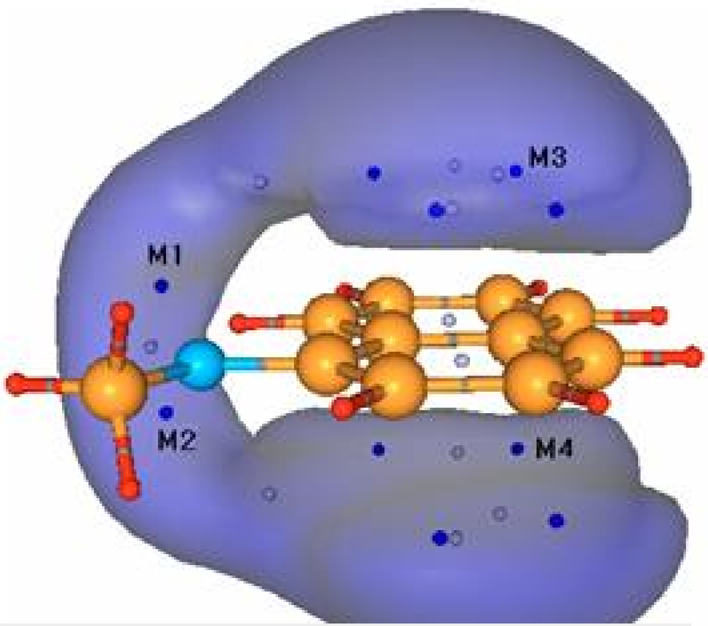
Computational Materials Modelling

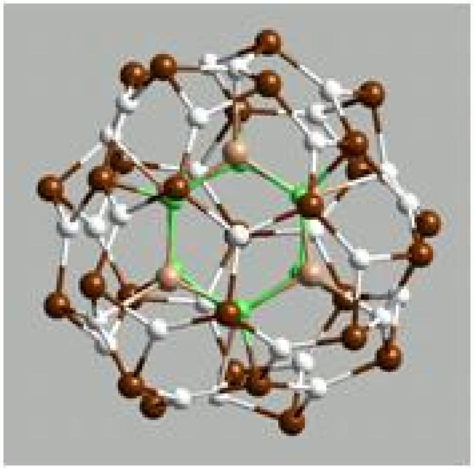
INDMOL suite of codes (Univ of Pune)

Study of Molecular Systems

Simulations of Quantum Dots







A quote to think

James Bailey

(New Era in Computation Ed. Metropolis & Rota)

We are all still trained to believe that orderly, sequential processes are more likely to be true. As we were reshaping our computers, so simultaneously were they reshaping us. May be when things happen in this world, they actually happen in parallel