Discrete Molecular Dynamics

Ensemble Computing with COMPSs

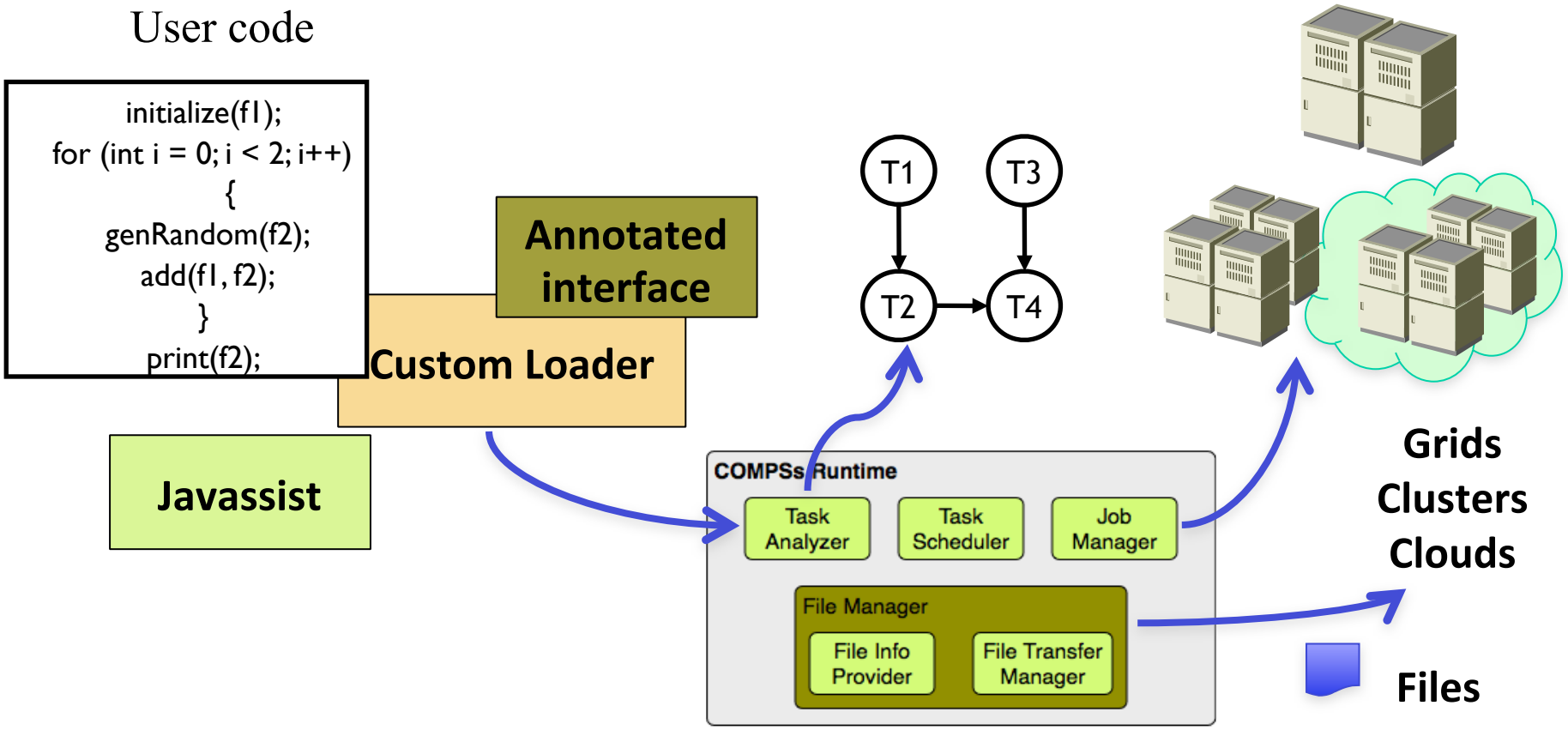
DISCRETE package

- Developed at BSC/IRB, Barcelona
- Implements the Discrete MD method
  - particles move until collision -> transfer momentum
  - no need for integration
  - event driven and time step corresponds to time between consecutive collisions
- Combined with multi-scale structure description offers fast docking protocols

COMPSs

- Reduces the development complexity of Grid/Cluster applications
- Suitable for applications that consist of repetitive tasks

COMPSs programming model



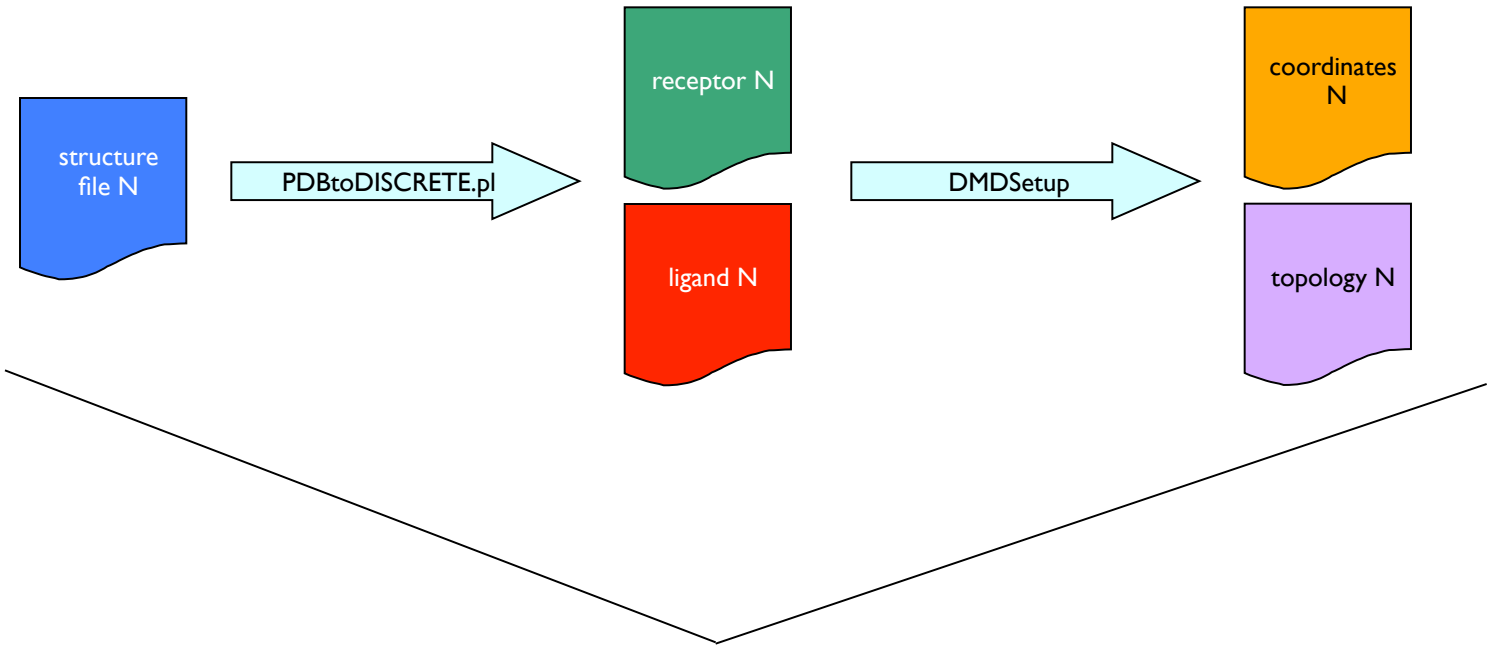
COMPSs highlights

- Transparent data management & task execution.
- Parallelization at task level.
- No application changes needed, sequential code remains unchanged.
- Good scalability.
- Based on standards.

Workflow definition

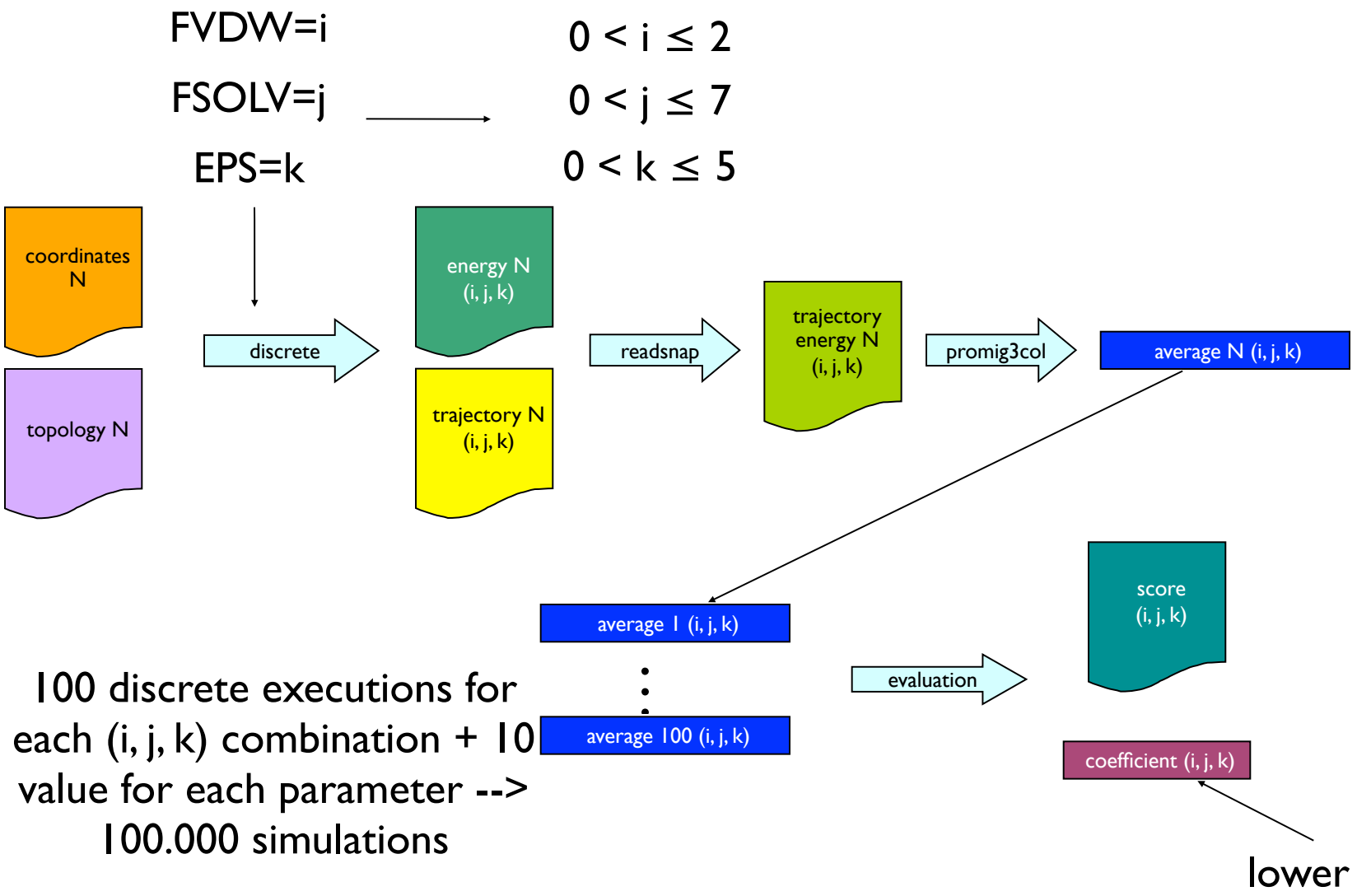
- Objective: launch multiple DISCRETE simulations to detect the optimal values for three input parameters
  - Use 100 structures as input cases
  - Use reference result to select cases to apply the score looking for minimum values
- Initial approach
  - No optimization, run all cases and compare results

Discrete Workflow (1)



once for each structure file (N=1..100)

Discrete Workflow (2)

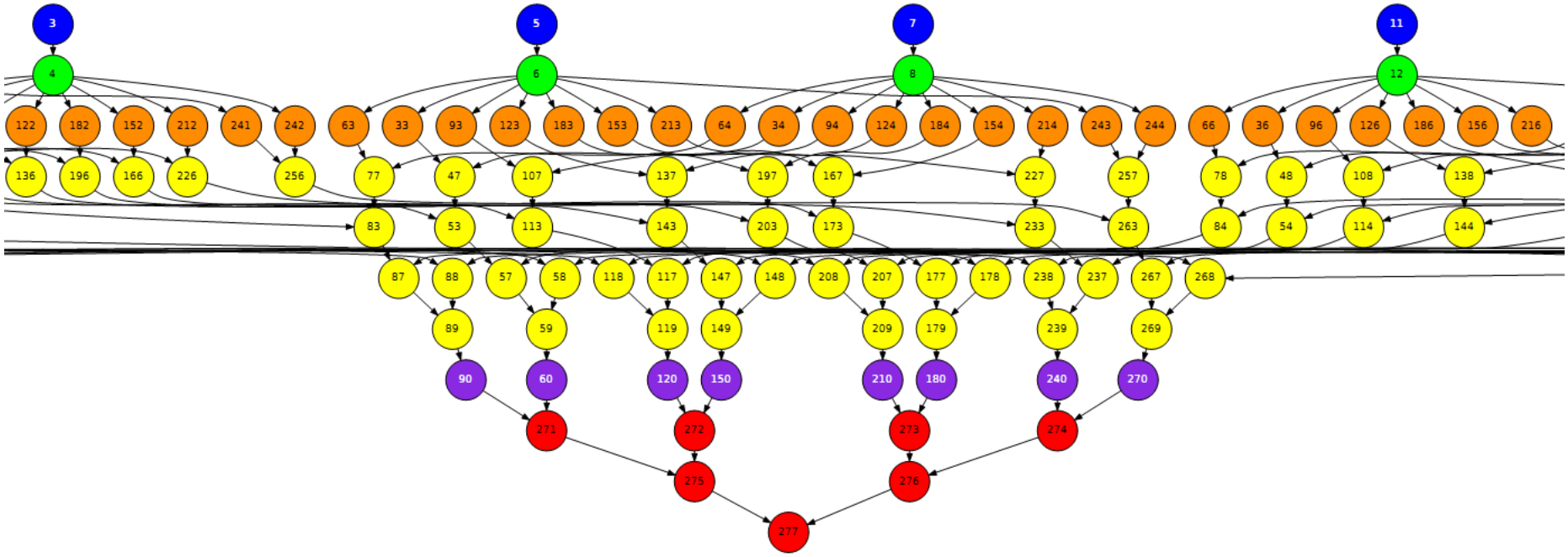


100 discrete executions for each (i, j, k) combination + 10 value for each parameter --> 100.000 simulations

Application Overview

- 6 methods (tasks):
  - **genReceptorLigand**: generates receptor and ligand given a structure file.
  - **dmdSetup**: runs DMDSetup given a receptor and a ligand.
  - **simulate**: runs discrete + readsnap + promig3col given a coordinates file and a topology + specific FVDW, FSOLV and EPS values. Returns average file.
  - **merge**: merges two average files into one. Used as part of a binary tree merging process to merge all averages into a single file.
  - **evaluate**: generates a score file and calculates the coefficient for all the averages and writes it to a file.
  - **min**: similar to merge, takes two coefficient files and returns the lower. Used in another binary tree to find the lowest coefficient.

Task Graph



genReceptorLigand

dmdSetup

simulate

merge

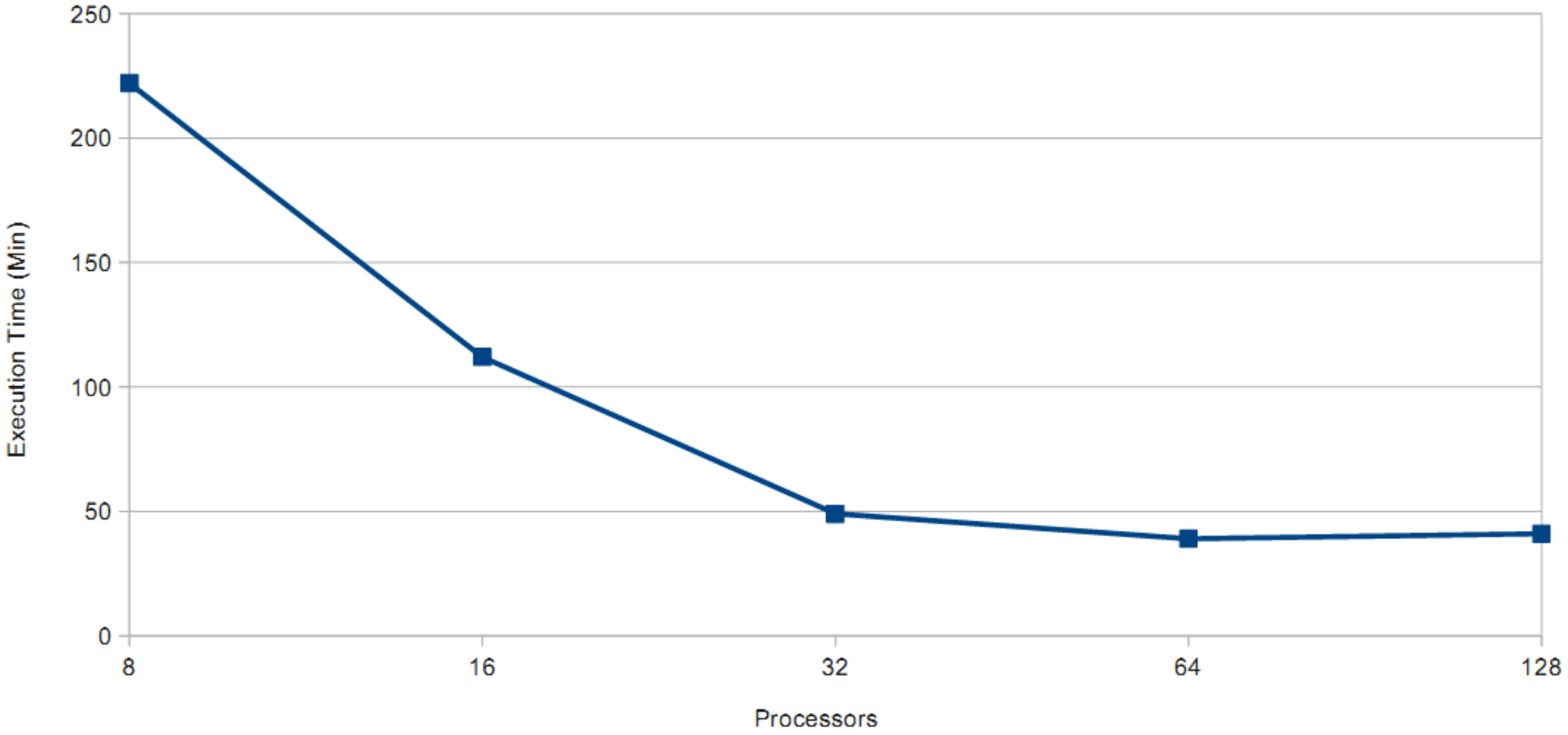
evaluate

min

Execution characteristics:

- 8 combinations of FVDW, FSOLV and EPS
- 10 simulations for each combination (80 in total)

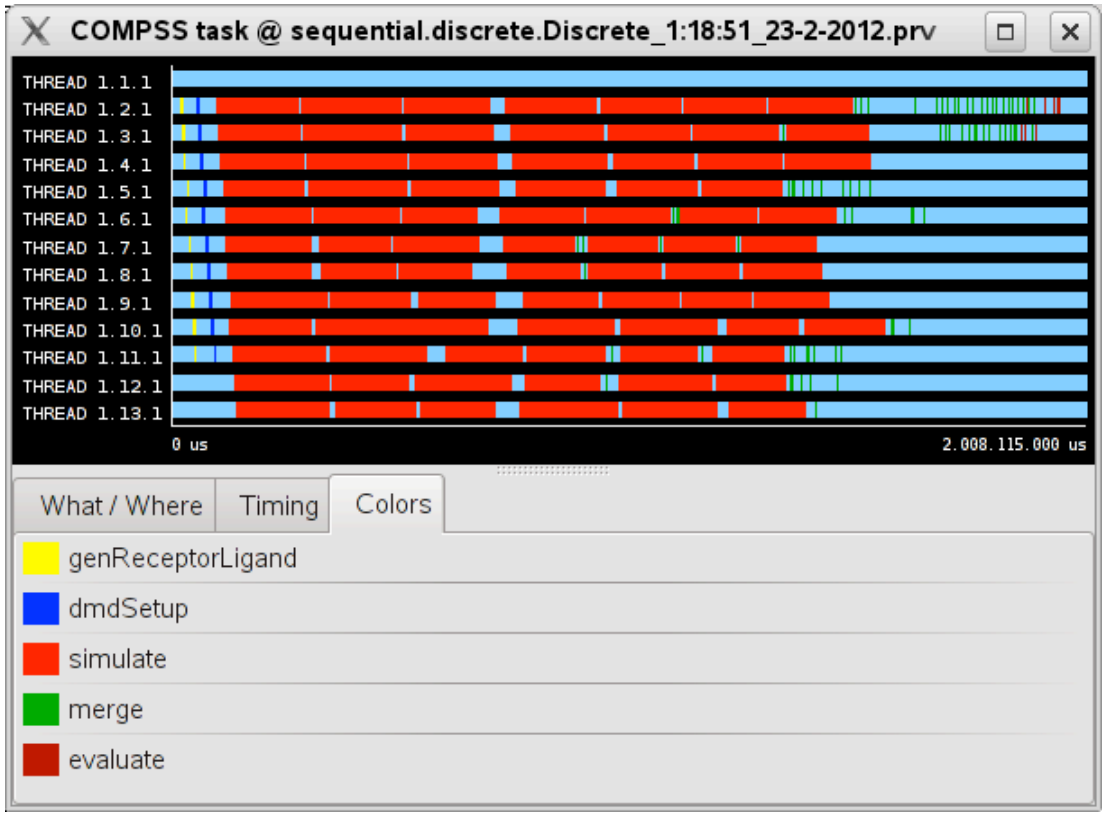
Scalability

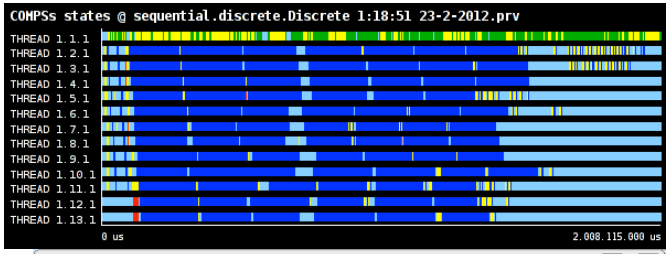


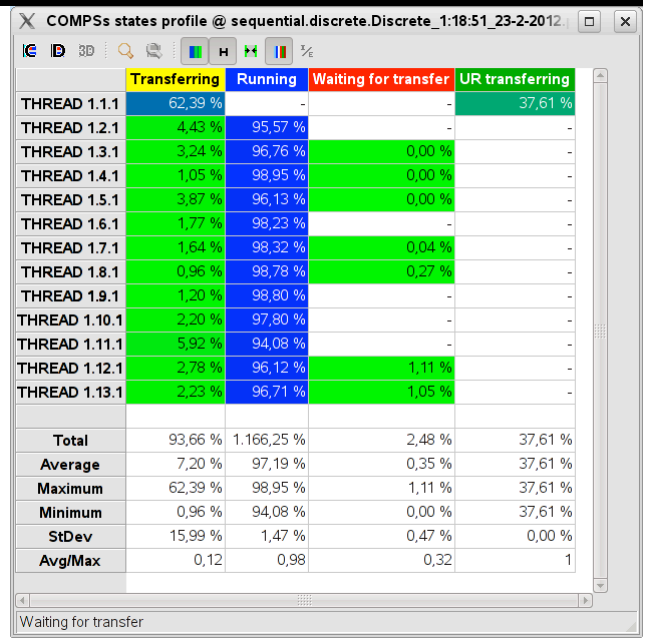
Execution characteristics:

- Run in MareNostrum
- 27 combinations of FVDW, FSOLV and EPS, 10 simulations for each combination (270 in total)

… and Paraver!







	Transferring	Running	Waiting for transfer	UR transferring
THREAD 1.1.1	62,39 %	-	-	37,61 %
THREAD 1.2.1	4,43 %	95,57 %	-	-
THREAD 1.3.1	3,24 %	96,76 %	0,00 %	-
THREAD 1.4.1	1,05 %	98,95 %	0,00 %	-
THREAD 1.5.1	3,87 %	96,13 %	0,00 %	-
THREAD 1.6.1	1,77 %	98,23 %	-	-
THREAD 1.7.1	1,64 %	98,32 %	0,04 %	-
THREAD 1.8.1	0,96 %	98,78 %	0,27 %	-
THREAD 1.9.1	1,20 %	98,80 %	-	-
THREAD 1.10.1	2,20 %	97,80 %	-	-
THREAD 1.11.1	5,92 %	94,08 %	-	-
THREAD 1.12.1	2,78 %	96,12 %	1,11 %	-
THREAD 1.13.1	2,23 %	96,71 %	1,05 %	-
Total	93,66 %	1.166,25 %	2,48 %	37,61 %
Average	7,20 %	97,19 %	0,35 %	37,61 %
Maximum	62,39 %	98,95 %	1,11 %	37,61 %
Minimum	0,96 %	94,08 %	0,00 %	37,61 %
StDev	15,99 %	1,47 %	0,47 %	0,00 %
Avg/Max	0,12	0,98	0,32	1

Execution characteristics:

- Run in MareNostrum with 12 computing processors
- 8 combinations of FVDW, FSOLV and EPS, 10 simulations for each combination (80 in total)

Acknowledgements

- DISCRETE: Agusti Emperador, Ramon Goni, Jose-Luis Gelpi (BSC), Modesto Orozco (IRB)
- COMPSs, Paraver: Judit Gimenez (BSC)

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

INSTITUTE FOR RESEARCH IN BIOMEDICINE

Annotated Interface (1)

```
public interface DiscreteItf {


        @Method(declaringClass = "discrete.DiscreteImpl")
    @Constraints(processorCPUCount = 1, memoryPhysicalSize = 1.5f)
                            void genReceptorLigand(
 @Parameter(type = Type.FILE, direction = Direction.IN) String pdbFile,
 @Parameter(type = Type.STRING, direction = Direction.IN) String binDir,
 @Parameter(type = Type.FILE, direction = Direction.OUT) String recFile,
@Parameter(type = Type.FILE, direction = Direction.OUT) String ligFile);


        @Method(declaringClass = "discrete.DiscreteImpl")
    @Constraints(processorCPUCount = 1, memoryPhysicalSize = 1.5f)
                            void dmdSetup(
 @Parameter(type = Type.FILE, direction = Direction.IN) String recFile,
 @Parameter(type = Type.FILE, direction = Direction.IN) String ligFile,
 @Parameter(type = Type.STRING, direction = Direction.IN) String binDir,
@Parameter(type = Type.STRING, direction = Direction.IN) String dataDir,
 @Parameter(type = Type.FILE, direction = Direction.OUT) String topFile,
@Parameter(type = Type.FILE, direction = Direction.OUT) String crdFile);


        @Method(declaringClass = "discrete.DiscreteImpl")
    @Constraints(processorCPUCount = 1, memoryPhysicalSize = 1.5f)
                            void simulate(
@Parameter(type = Type.FILE, direction = Direction.IN) String paramFile,
 @Parameter(type = Type.FILE, direction = Direction.IN) String topFile,
 @Parameter(type = Type.FILE, direction = Direction.IN) String crdFile,
 @Parameter(type = Type.STRING, direction = Direction.IN) String natom,
 @Parameter(type = Type.STRING, direction = Direction.IN) String binDir,
@Parameter(type = Type.STRING, direction = Direction.IN) String dataDir,
@Parameter(type = Type.FILE, direction = Direction.OUT) String average);
```

Annotated Interface (2)

```
        @Method(declaringClass = "discrete.DiscreteImpl")
    @Constraints(processorCPUCount = 1, memoryPhysicalSize = 1.5f)
                            void merge(
    @Parameter(type = Type.FILE, direction = Direction.INOUT) String r1,
     @Parameter(type = Type.FILE, direction = Direction.IN) String r2);

        @Method(declaringClass = "discrete.DiscreteImpl")
    @Constraints(processorCPUCount = 1, memoryPhysicalSize = 1.5f)
                            void evaluate(
@Parameter(type = Type.FILE, direction = Direction.IN) String averageFile,
@Parameter(type = Type.FILE, direction = Direction.IN) String pydockFile,
  @Parameter(type = Type.STRING, direction = Direction.IN) String fvdw,
  @Parameter(type = Type.STRING, direction = Direction.IN) String fsolv,
   @Parameter(type = Type.STRING, direction = Direction.IN) String eps,
@Parameter(type = Type.FILE, direction = Direction.OUT) String scoreFile,
 @Parameter(type = Type.FILE, direction = Direction.OUT) String coeffFile

                            );


        @Method(declaringClass = "discrete.DiscreteImpl")
    @Constraints(processorCPUCount = 1, memoryPhysicalSize = 1.5f)
                            void min(
    @Parameter(type = Type.FILE, direction = Direction.INOUT) String f1,
     @Parameter(type = Type.FILE, direction = Direction.IN) String f2);
                            }
```

Method Invocations

```
// generate receptor and ligand and run setup
for (int i = 1; i <= N; i++) {
String pdbFile = "/1B6C_" + i + ".pdb";
String recFile = "/receptor_" + i;
…
DiscreteImpl.genReceptorLigand(pdbFile, binDir, recFile, ligFile);
DiscreteImpl.dmdSetup(recFile, ligFile, binDir, dataDir, topFile, crdFile);
}
…
// parameter sweep loops
for (int i = 1; i <= STEPS; i++) {
for (int j = 1; j <= STEPS; j++) {
for (int k = 1; k <= STEPS; k++) {
…
// run the N simulations
for (int ii = 1; ii <= N; ii++) {
…  DiscreteImpl.simulate(paramFile, topFile, crdFile, … , averageFile);
list.add(averageFile);
}

// merge all averages in a single file using a binary tree process
while (…) {
String r1 = list.poll();
String r2 = list.poll();
DiscreteImpl.merge(r1, r2);
}
// evaluate score and write coefficient
DiscreteImpl.evaluate(list.poll(), … , scoreFile, coeffFile);
coeffList.add(coeffFile);
}
}
}
// find out the min coefficient of all configurations using another binary tree
while (coeffList.size() > 1) {
String c1 = coeffList.poll();
DiscreteImpl.min(c1, coeffList.poll());
coeffList.add(c1);   //at the end the last file in the list is the minimum
}
```