UT Southwestern Medical Center
Lyda Hill Department of Bioinformatics

BioHPC

Basics of Linux III

Examples

[web] portal.biohpc.swmed.edu
[email] biohpc-help@utsouthwestern.edu

2018-05-16

Working with Functions

```
➢ cat para.txt
dose 9.8
pixel 15.8
cancel
```

```
#!/bin/bash

function readpar()
{
   local var=`grep -i "$tmp" "$par" | awk '{print $2;}'`;
   echo "$var"
}

dose="1.6"
pixel="0.415"
par="para.txt"
grep 2>/dev/null -q -i 'pixel' "$par" && { tmp="pixel"; pixel=$(readpar); }
grep 2>/dev/null -q -i 'dose' "$par" && { tmp="dose"; dose=$(readpar); }

echo "pixel : "$pixel
echo "dose : "$dose
```

Working with Functions

```
➢ cat para.txt
dose 9.8
pixel 15.8
cancel
```

```
#!/bin/bash

now() { date "+%Y-%m-%d %H:%M:%S.%3N"; }
iscancelled() {
        if [ -s para.txt ]
        then
                if grep -q -i 'cancel' para.txt;
                then
                        return 0
                fi
        fi
        return 1
}
function readpar()
{
   local var=`grep -i "$tmp" "$par" | awk '{print $2;}'`;
   echo "$var"
}
```

```
dose="1.6"
pixel="0.415"
par="para.txt"

while true
do
        echo "Looping !!!"
        if iscancelled $dir; then
           printf "%s : cancelled by para.txt in, Exiting %s\n" "$(now)"
           exit 1
        fi
        grep 2>/dev/null -q -i 'pixel' "$par" && { tmp="pixel"; pixel=$(readpar); }
        grep 2>/dev/null -q -i 'dose' "$par" && { tmp="dose"; dose=$(readpar); }
        echo "pixel : "$pixel
        echo "dose : "$dose
        mkdir -p pixel-$pixel
        mkdir 2>/dev/null dose-$dose
        sleep 5
Done
```

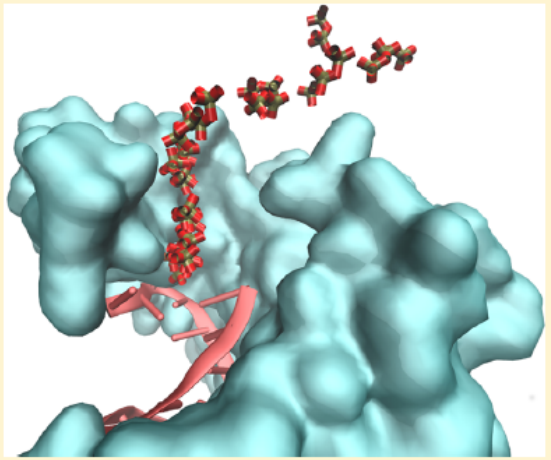
A problem

Pyrophosphate Release in the Protein HIV Reverse Transcriptase

Murat Atis,[†] Kenneth A. Johnson,[‡] and Ron Elber[*,†,§]

[†]Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, Texas 78712, United States
[‡]Department of Molecular Biosciences, The University of Texas at Austin, Austin, Texas 78712, United States
[§]Department of Chemistry, The University of Texas at Austin, Austin, Texas 78712, United States

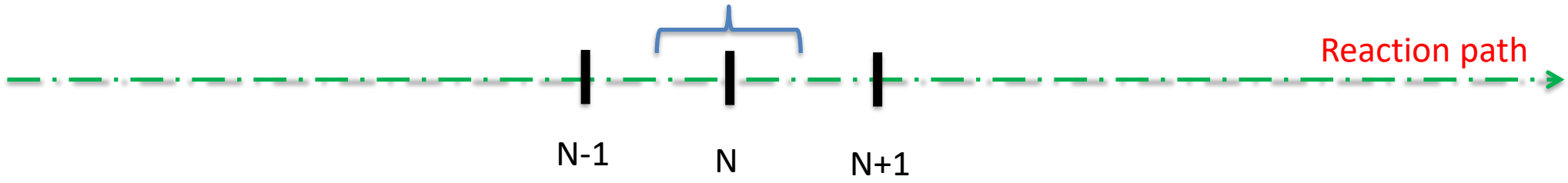
ABSTRACT: Enzymatic reactions usually occur in several steps: a step of substrate binding to the surface of the protein, a step of protein reorganization around the substrate and conduction of a chemical reaction, and a step of product release. The release of inorganic phosphate—PPi—from the matrix of the protein HIV reverse transcriptase is investigated computationally. Atomically detailed simulations with explicit solvent are analyzed to obtain the free energy profile, mean first passage time, and detailed molecular mechanisms of PPi escape. A challenge for the computations is of time scales. The experimental time scale of the process of interest is in milliseconds, and straightforward molecular dynamics simulations are in sub-microseconds. To overcome the time scale gap, we use the algorithm of Milestoning along a reaction coordinate to compute the overall free energy profile and rate. The methods of locally enhanced sampling and steered molecular dynamics determine plausible reaction coordinates. The observed molecular mechanism couples the transfer of the PPi to positively charged lysine side chains that are found on the exit pathway and to an exiting magnesium ion. In accord with experimental findings, the release rate is comparable to the chemical step, allowing for variations in substrate (DNA or RNA template) in which the release becomes rate determining.



Long time for ms level simulation.

A solution & another problem

Milestoning ! But required lots of different calculation steps.



100-500 samples for each milestone (69 milestones total) and long molecular dynamics simulation testing for each of sample. Checking passed previous or next milestone and count them. Thousands of calculations and analysis!

Shell commands and tools

```
login1.ls5(450)$ head test-23/try-23-1/output.colvars.traj
      # step         ddpo
         0    5.33229669473108e+01
        10    5.33193990641006e+01
        20    5.33122097489066e+01
        30    5.33079687575632e+01
        40    5.33015689177682e+01
        50    5.32905088759830e+01
        60    5.32759593328351e+01
        70    5.32655676532990e+01
        80    5.32527725868597e+01
```

← Find passing point at thousands file by one by

Some Examples

```
➢ cat stopthem
ps ux | grep "namd2" | awk '{system("kill -9 "$2);}'
```

```
➢ head test-23/try-23-1/output.colvars.traj
 # step         ddpo
   0    5.33229669473108e+01
  10    5.33193990641006e+01
  20    5.33122097489066e+01
  30    5.33079687575632e+01
  40    5.33015689177682e+01  ← Find passing point
  50    5.32905088759830e+01     at thousands file by
  60    5.32759593328351e+01     one by
```

```
➢ cat stopper
#!/bin/bash
for (( c=1; c<=5640; c++ ))
do
        sleep 30
        awk -v ll=43.00 '{if((($2>=ll+0.5) || ($2<=ll-0.50)) && ($1 != "#") && ($2 != "")) {
                print $0; system("./stopthem"); exit 1}}' output.colvars.traj
done
```

```
➢ cat check
#!/bin/bash
for (( c=1; c<=100; c++ ))
do
        awk -v ll=43.00 '{if(($2>=ll+0.50) && ($1 != "#") && ($2 != "")) {print "next", $0; exit 1}
            else if(($2<=ll-0.50) && ($1 != "#") && ($2 != "")) {print "prev", $0; exit 1}}' try-01-$c/output.colvars.traj
done
```

Some Examples

```
➢ cat stopthem
ps ux | grep "namd2" | awk '{system("kill -9 "$2);}'
```

```
➢ head test-23/try-23-1/output.colvars.traj
 # step         ddpo
   0    5.33229669473108e+01
  10    5.33193990641006e+01
  20    5.33122097489066e+01
  30    5.33079687575632e+01
  40    5.33015689177682e+01  ← Find passing point
  50    5.32905088759830e+01     at thousands file by
  60    5.32759593328351e+01     one by
```

```
➢ cat stopper
#!/bin/bash
for (( c=1; c<=5640; c++ ))
do
        sleep 30
        awk -v ll=43.00 '{if((($2>=ll+0.5) || ($2<=ll-0.50)) && ($1 != "#") && ($2 != "")) {
                print $0; system("./stopthem"); exit 1}}' output.colvars.traj
done
```

```
➢ cat check
#!/bin/bash
for (( c=1; c<=100; c++ ))
do
        awk -v ll=43.00 '{if(($2>=ll+0.50) && ($1 != "#") && ($2 != "")) {print "next", $0; exit 1}
            else if(($2<=ll-0.50) && ($1 != "#") && ($2 != "")) {print "prev", $0; exit 1}}' try-01-$c/output.colvars.traj
done
```

Slurm script example

```
#!/bin/bash
#SBATCH --job-name="test-01"
#SBATCH --output="namd.%j.%N.out"
#SBATCH --partition=normal
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=16
#SBATCH --export=ALL
#SBATCH -A A-ices9
#SBATCH -t 47:00:00

module load namd/2.10
cd /work/04394/atis/LES3/run-SMD2/test-01
killall stopper
./stopper &
for (( c=1; c<=100; c++ ))
do
        mkdir try-01-$c
        #Copy the target pdb
        cp ../m01/sample-$c.pdb mile.pdb
        #Run MPI job using ibrun
        ibrun namd2 in.namd > output.log

        rm -f output.restart.*
        mv output.* try-01-$c
done
killall stopper
```

Preparing inputs

```
login1.ls5(390)$ ls
      sampletest
login1.ls5(400)$ ls -1 sampletest/
      stopper
      check
      colvars.in
      in.namd
      mile.psf
      Mile.pdb
      job-submit
      stopthem
login1.ls5(401)$ for i in $(seq –w 01 40); do  cp -r sampletest test-$i;   done;
login1.ls5(402)$ ls
      Sampletest  test-03  test-06  test-09  test-12  test-15  test-18  test-21  test-24  test-27  test-30  test-33  test-36  test-39
      test-01     test-04  test-07  test-10  test-13  test-16  test-19  test-22  test-25  test-28  test-31  test-34  test-37  test-40
      test-02     test-05  test-08  test-11  test-14  test-17  test-20  test-23  test-26  test-29  test-32  test-35  test-38
```

colvars.in

```
colvarsTrajFrequency 10

colvar {
   name ddpo

   distance {
      group1 { atomNumbersRange 16192-17691 }
      group2 { atomNumbersRange 16181-16189 }
   }

   lowerBoundary 0
   upperBoundary 75
}
```

job-submit

```
#!/bin/bash
#SBATCH --job-name="test-01"
#SBATCH --output="namd.%j.%N.out"
#SBATCH --partition=normal
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=16
#SBATCH --export=ALL
#SBATCH -A A-ices9
#SBATCH -t 47:00:00

module load namd/2.10
cd /work/04394/atis/LES3/run-SMD2/test-01
killall stopper
./stopper &
for (( c=1; c<=100; c++ ))
do
mkdir try-01-$c

#Copy the target pdb
cp ../m#1#/sample-$c.pdb mile.pdb
#Run MPI job using ibrun
ibrun namd2 in.namd > output.log

rm -f output.restart.*
mv output.* try-01-$c

done
killall stopper
```

stopper

```
#!/bin/bash
for (( c=1; c<=5640; c++ ))
do
sleep 30
awk -v ll=43.00 '{if((($2>=ll+0.5) || ($2<=ll-0.50)) && ($1 != "#") && ($2 != "")) {print $0; system("./stopthem"); exit 1}} ' output.colvars.traj
done
```

```
ps ux | grep "namd2" | awk '{system("kill -9 "$2);}'
```

stopthem

check

```
#!/bin/bash
for (( c=1; c<=100; c++ ))
do
awk -v ll=43.00 '{if(($2>=ll+0.50) && ($1 != "#") && ($2 != "")) {print "next", $0; exit 1} else if(($2<=ll-0.50) && ($1 != "#") && ($2 != "")) {print "prev", $0; exit 1}}'
try-01-$c/output.colvars.traj
done
```

colvars.in

```
colvarsTrajFrequency 10

colvar {
   name ddpo

   distance {
      group1 { atomNumbersRange 16192-17691 }
      group2 { atomNumbersRange 16181-16189 }
   }

   lowerBoundary 0
   upperBoundary 75
}
```

job-submit

```
#!/bin/bash
#SBATCH --job-name="test-#1#"
#SBATCH --output="namd.%j.%N.out"
#SBATCH --partition=normal
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=16
#SBATCH --export=ALL
#SBATCH -A A-ices9
#SBATCH -t 47:00:00

module load namd/2.10
cd /work/04394/atis/LES3/run-SMD2/test-#1#
killall stopper
./stopper &
for (( c=1; c<=100; c++ ))
do
mkdir try-#1#-$c

#Copy the target pdb
cp ../m#1#/sample-$c.pdb mile.pdb
#Run MPI job using ibrun
ibrun namd2 in.namd > output.log

rm -f output.restart.*
mv output.* try-#1#-$c

done
killall stopper
```

stopper

```
#!/bin/bash
for (( c=1; c<=5640; c++ ))
do
sleep 30
awk -v ll=#2# '{if((($2>=ll+0.5) || ($2<=ll-0.50)) && ($1 != "#") && ($2 != "")) {print $0; system("./stopthem"); exit 1}} ' output.colvars.traj
done
```

stopthem

```
ps ux | grep "namd2" | awk '{system("kill -9 "$2);}'
```

check

```
#!/bin/bash
for (( c=1; c<=100; c++ ))
do
awk -v ll=#2# '{if(($2>=ll+0.50) && ($1 != "#") && ($2 != "")) {print "next", $0; exit 1} else if(($2<=ll-0.50) && ($1 != "#") && ($2 != "")) {print "prev", $0; exit 1}}'
try-#1#-$c/output.colvars.traj
done
```

```
for i in $(seq –w 01 40); do  ii=$(echo "43.00+($i-1)*0.5" | bc -l) ; cp -r sampletest test-$i; sed -i 's/#2#/'$ii'/g' test-$i/check; sed -i 's/#2#/'$ii'/g' test-$i/stopper; sed -i 's/#1#/'$i'/g' test-$i/check; sed -i 's/#1#/'$i'/g' test-$i/job-submit; done;
```

OR

```
for i in $(seq -w 01 40); do
        ii=$(echo "43.00+($i-1)*0.5" | bc -l)
        cp -r sampletest test-$i
        sed -i 's/#2#/'$ii'/g' test-$i/check
        sed -i 's/#2#/'$ii'/g' test-$i/stopper
        sed -i 's/#1#/'$i'/g' test-$i/check;
        sed -i 's/#1#/'$i'/g' test-$i/job-submit
done
```

Starting calculation

```
login1.ls5(447)$ for i in $(seq –w 01 40); do  sbatch test-$i/job-submit ; done;
…
login1.ls5(450)$ head test-23/try-23-1/output.colvars.traj
         # step         ddpo
         0    5.33229669473108e+01
        10    5.33193990641006e+01
        20    5.33122097489066e+01
        30    5.33079687575632e+01
        40    5.33015689177682e+01
        50    5.32905088759830e+01
        60    5.32759593328351e+01
        70    5.32655676532990e+01
        80    5.32527725868597e+01
```

Analyzing

```
login1.ls5(447)$ for i in $(seq –w 01 40); do test-$i/check > passing.txt ; done;
login1.ls5(451)$ head test-23/passing.txt
      prev       10540    5.34824965415358e+01
      next          90    5.45203997443408e+01
login1.ls5(467)$ for i in $(seq –w 01 40); do  grep next test-$i/passing.txt | wc -l >> next.txt; done;
login1.ls5(467)$ for i in $(seq –w 01 40); do  grep prev test-$i/passing.txt | wc -l >> prev.txt; done;
login1.ls5(467)$ for i in $(seq –w 01 40); do  wc -l test-$i/passing.txt >> tot.txt;  done;
login1.ls5(469)$ paste prev.txt next.txt tot.txt > combine.txt
login1.ls5(470)$ head combine.txt
      62 38 100 test-01/passing.txt
      84 16 100 test-02/passing.txt
      88 12 100 test-03/passing.txt
      83 17 100 test-04/passing.txt
      86 14 100 test-05/passing.txt
```

Thanks !