Recent Advancements in Adaptive Kinetic Monte Carlo

Samuel T. Chill and Graeme Henkelman

Henkelman Group
Department of Chemistry
The University of Texas at Austin

June 10, 2013

Overview

Background: Intoduction to Adaptive Kinetic Monte Carlo (AKMC)

Two Improvements to AKMC:
Part I: Improved saddle searches using molecular dynamics

- Compare effiency to min-mode following searches
- Estimate the uncertainty in the rate table

Part II: Treating superbasins using absorbing Markov chains

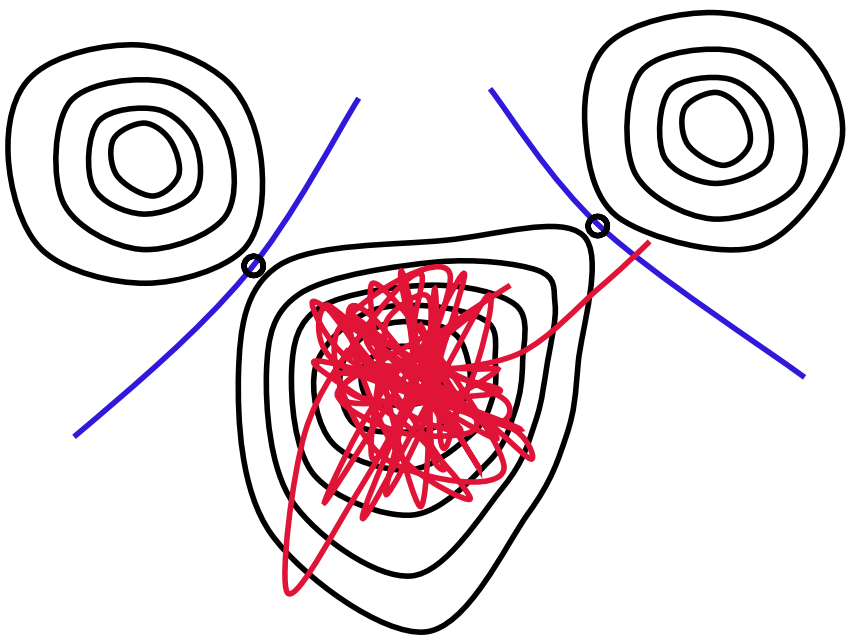
- When isn't KMC fast enough
- Monte Carlo with absorbing Markov Chains (MCAMC)

Problem Description

Rare Event System

A chemical system where the atoms spend large amount of time in each energy basin before transitioning to the next.

- Cannot use molecular dynamics



Reproduced from Art Voter's "Introduction to The Kinetic Monte Carlo Method"

How to efficiency model the state-to-state dynamics?

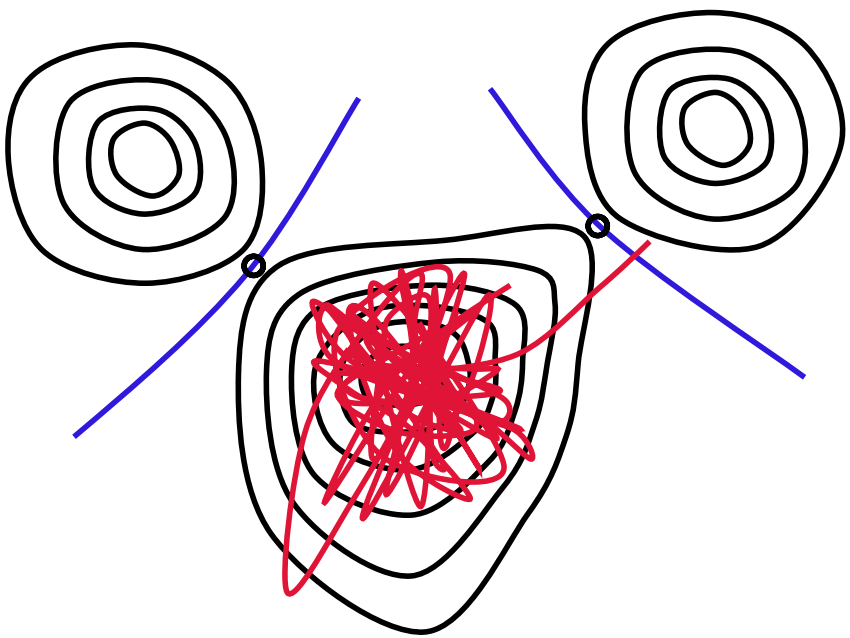
- Parallelize over time
- Accelerated Dynamics
  - Alter potential energy surface
  - Increase temperature
- Statistical mechanics

Problem Description

Rare Event System

A chemical system where the atoms spend large amount of time in each energy basin before transitioning to the next.

- Cannot use molecular dynamics



Reproduced from Art Voter's "Introduction to The Kinetic Monte Carlo Method"

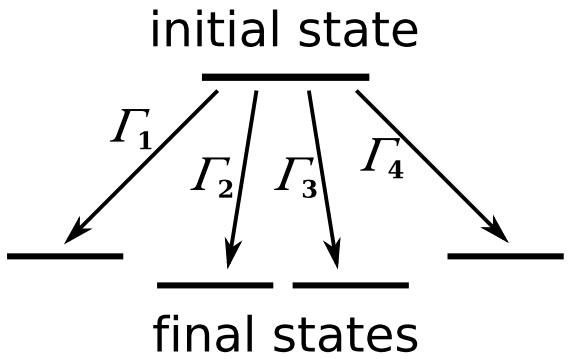
How to efficiency model the state-to-state dynamics?

- Parallelize over time
- Accelerated Dynamics
  - Alter potential energy surface
  - Increase temperature
- Statistical mechanics

Kinetic Monte Carlo

Models state-to-state dynamics as a Markov chain

- The states must be Markovian
- Next event is chosen in proportion to its rate
- Escape time drawn from exponential distribution



$$P\left[\Delta t\right] = \exp\left(-\Delta t \sum_i \Gamma_i\right)$$

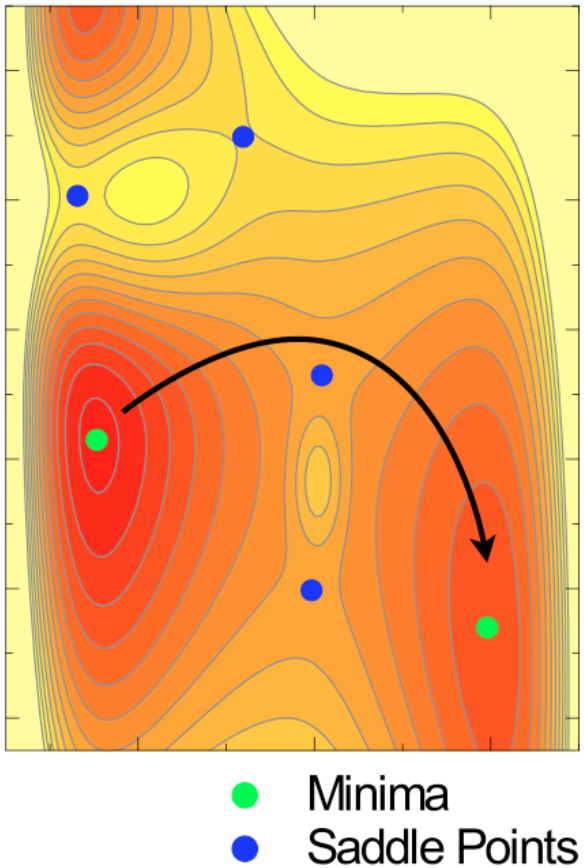
- Very fast (two random numbers and some book keeping)

How to build the Markov model? How to get the states and the rates?

Adaptive Kinetic Monte Carlo

Algorithm

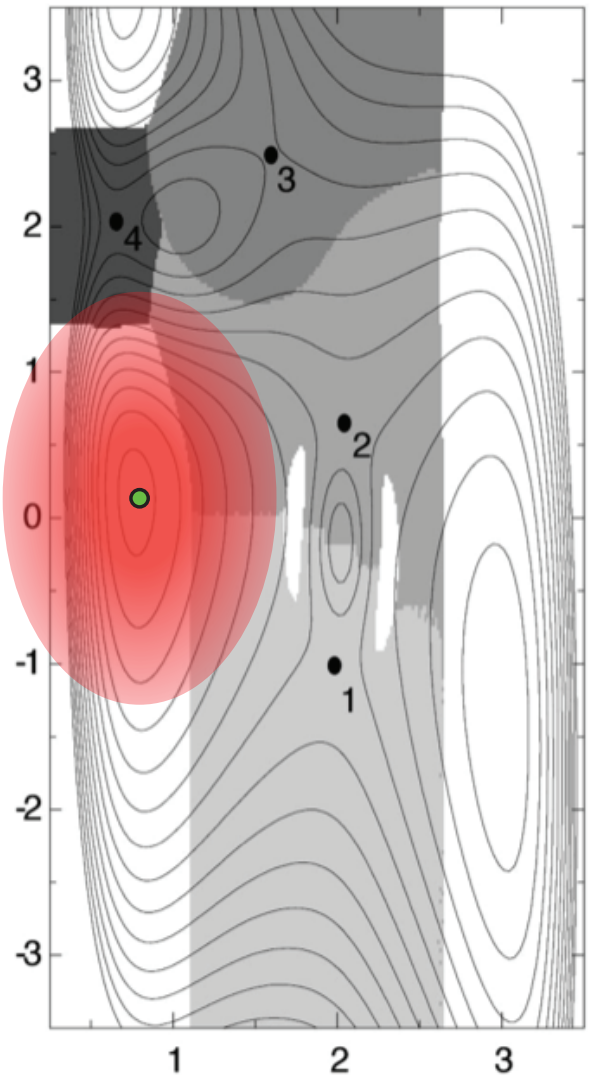
1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



Adaptive Kinetic Monte Carlo

Algorithm

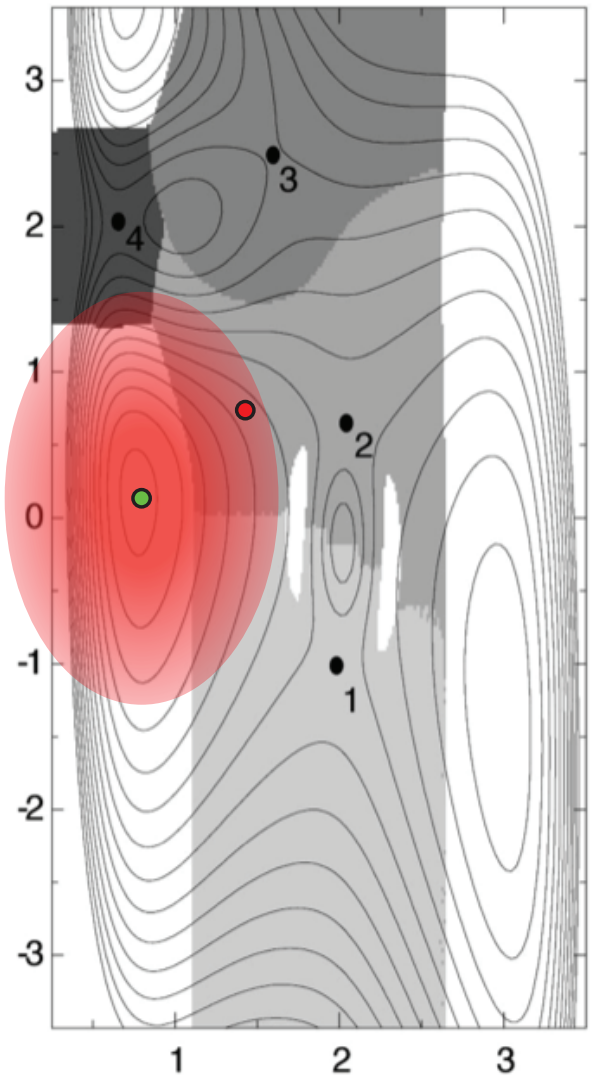
1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



Adaptive Kinetic Monte Carlo

Algorithm

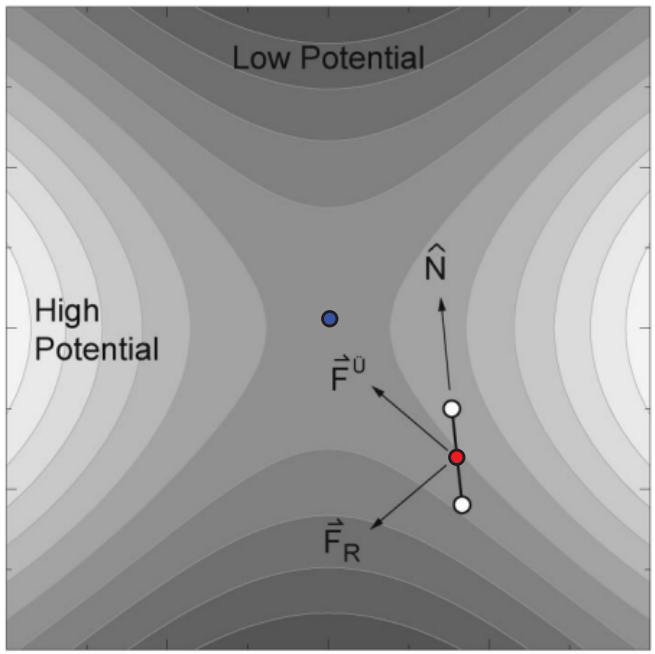
1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



Adaptive Kinetic Monte Carlo

Algorithm

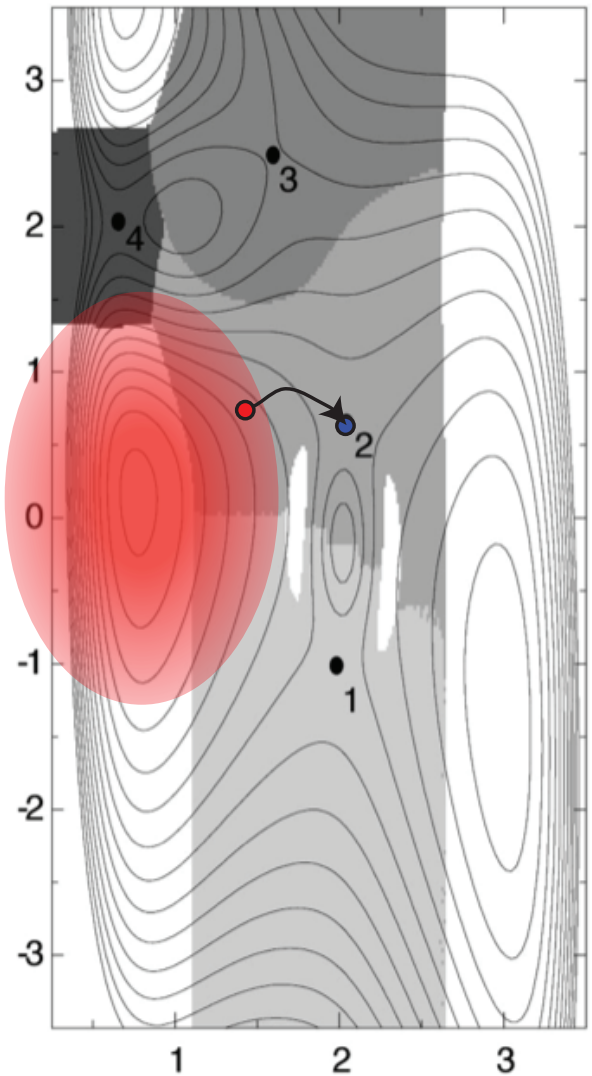
1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



Adaptive Kinetic Monte Carlo

Algorithm

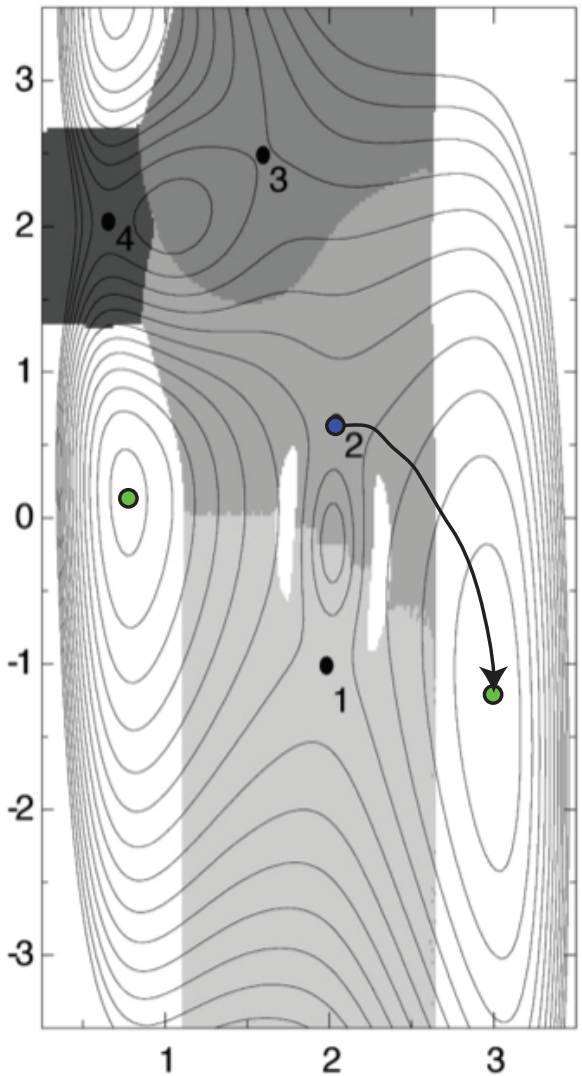
1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



Adaptive Kinetic Monte Carlo

Algorithm

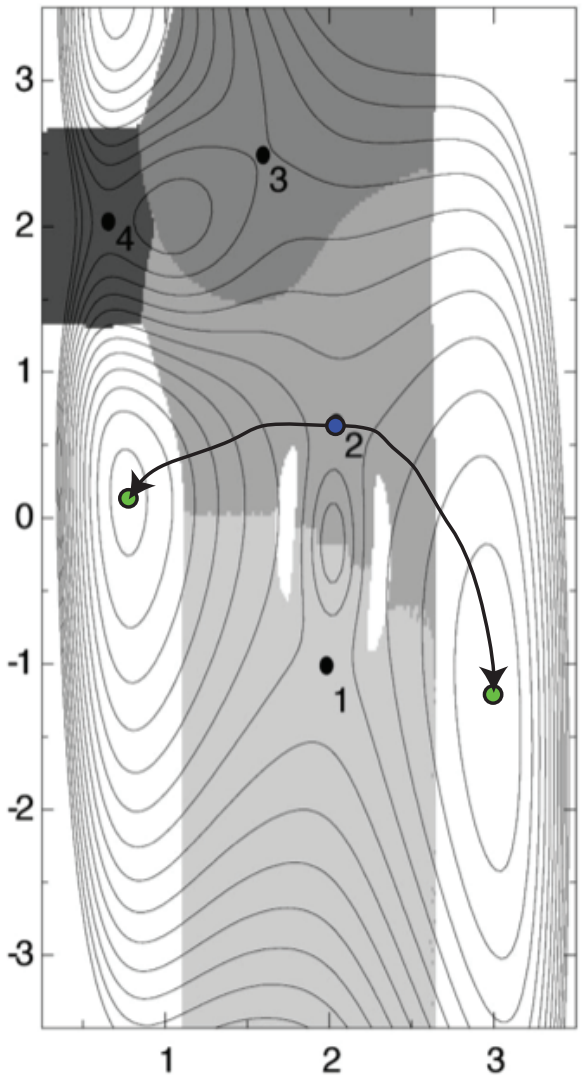
1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



Adaptive Kinetic Monte Carlo

Algorithm

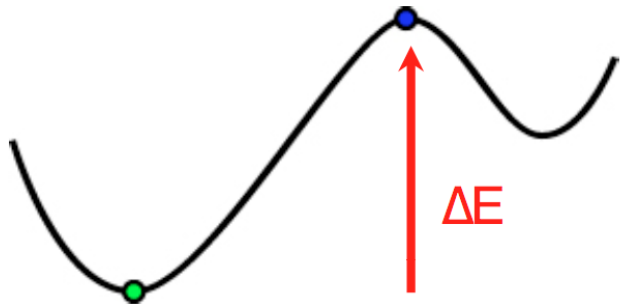
1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



Adaptive Kinetic Monte Carlo

Algorithm

1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



$$r = A \exp\left[-\Delta E / k_{\mathrm{B}} T\right]$$

Adaptive Kinetic Monte Carlo

Algorithm

1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached

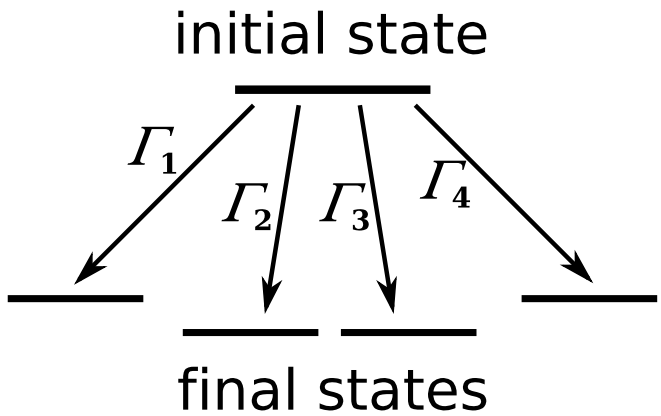
$$C = 1 - \frac{1}{N}$$

N is number of consecutive searches that did not find a new unique saddle

Adaptive Kinetic Monte Carlo

Algorithm

1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached



Adaptive Kinetic Monte Carlo

Algorithm

1. Displace randomly from the current minimum
2. Min-mode saddle search (using dimer or Lanczos)
3. Minimize along negative curvature at saddle to find new product state
4. Ensure connectivity
5. Calculate rate using HTST
6. Estimate confidence that all important saddles have been found
7. Take KMC steps until a new state is reached

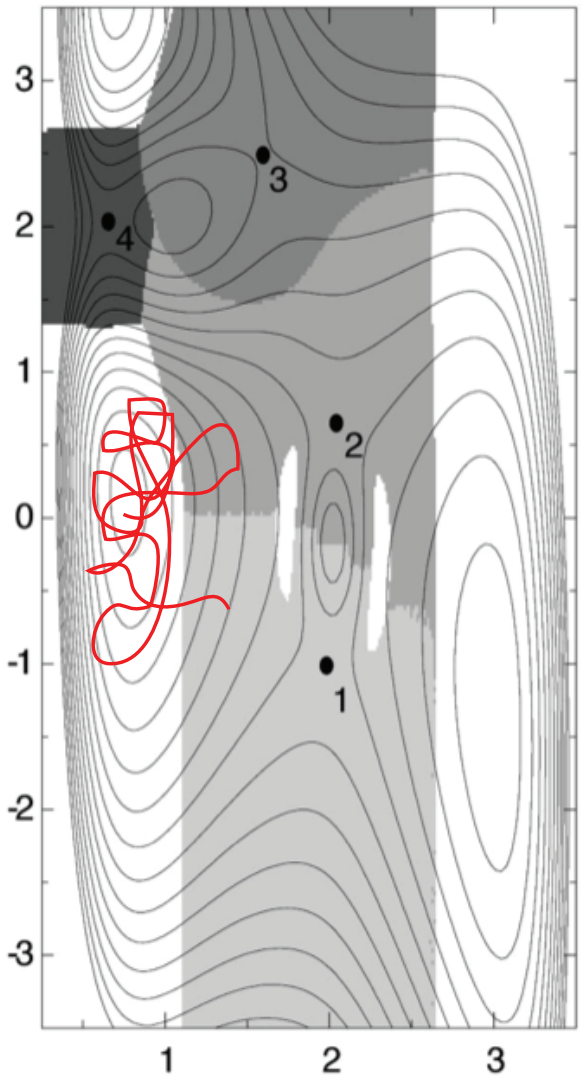
Use high temperature MD to generate displacements.

Using High Temperature Molecular Dynamics to Find Saddles

Replace random displacements with MD displacements.

Algorithm

1. Run high temperature MD
2. Periodically minimize the system to see if it has exited
3. Connect the initial minimum to the product minimum via a chain-of-states method such as Nudged Elastic Band
4. Find first maxima along the minimized band

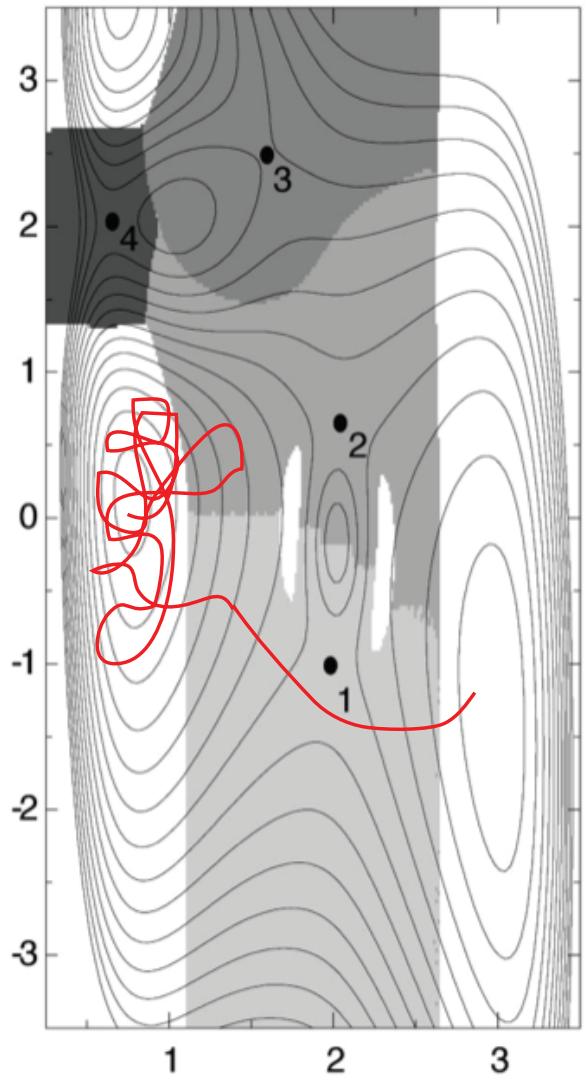


Using High Temperature Molecular Dynamics to Find Saddles

Replace random displacements with MD displacements.

Algorithm

1. Run high temperature MD
2. Periodically minimize the system to see if it has exited
3. Connect the initial minimum to the product minimum via a chain-of-states method such as Nudged Elastic Band
4. Find first maxima along the minimized band

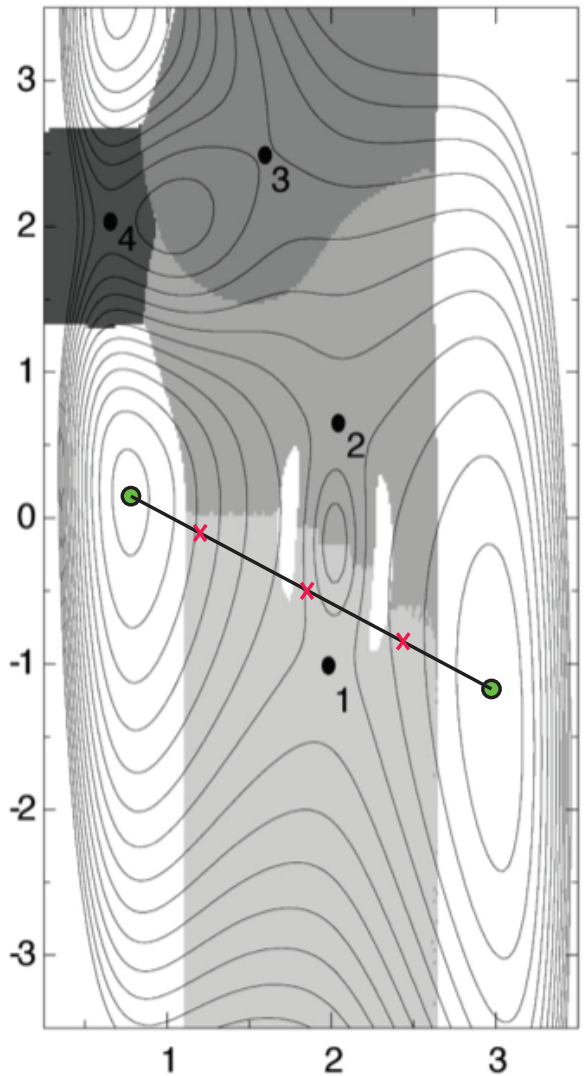


Using High Temperature Molecular Dynamics to Find Saddles

Replace random displacements with MD displacements.

Algorithm

1. Run high temperature MD
2. Periodically minimize the system to see if it has exited
3. Connect the initial minimum to the product minimum via a chain-of-states method such as Nudged Elastic Band
4. Find first maxima along the minimized band

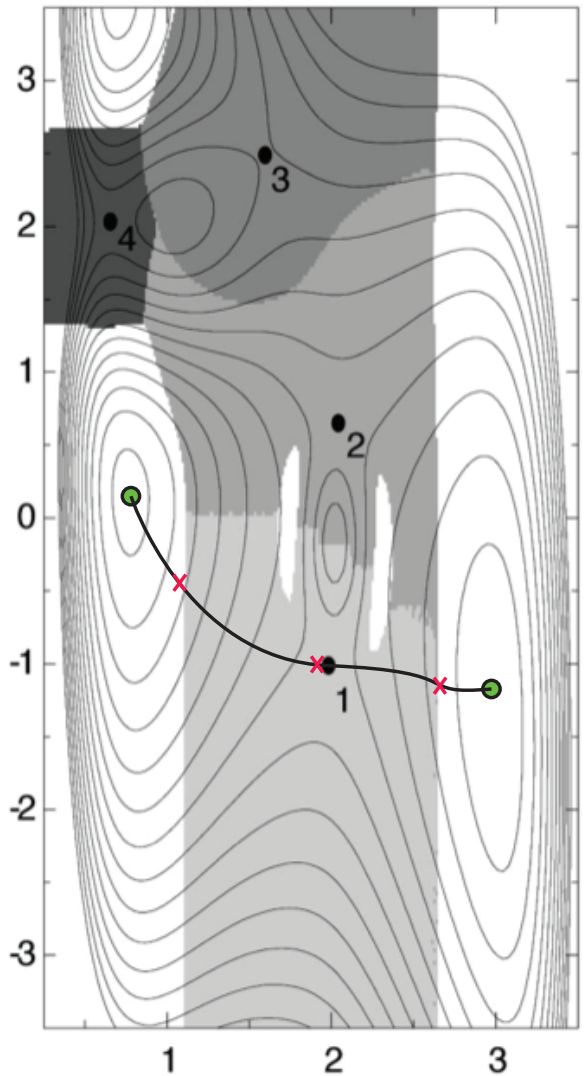


Using High Temperature Molecular Dynamics to Find Saddles

Replace random displacements with MD displacements.

Algorithm

1. Run high temperature MD
2. Periodically minimize the system to see if it has exited
3. Connect the initial minimum to the product minimum via a chain-of-states method such as Nudged Elastic Band
4. Find first maxima along the minimized band

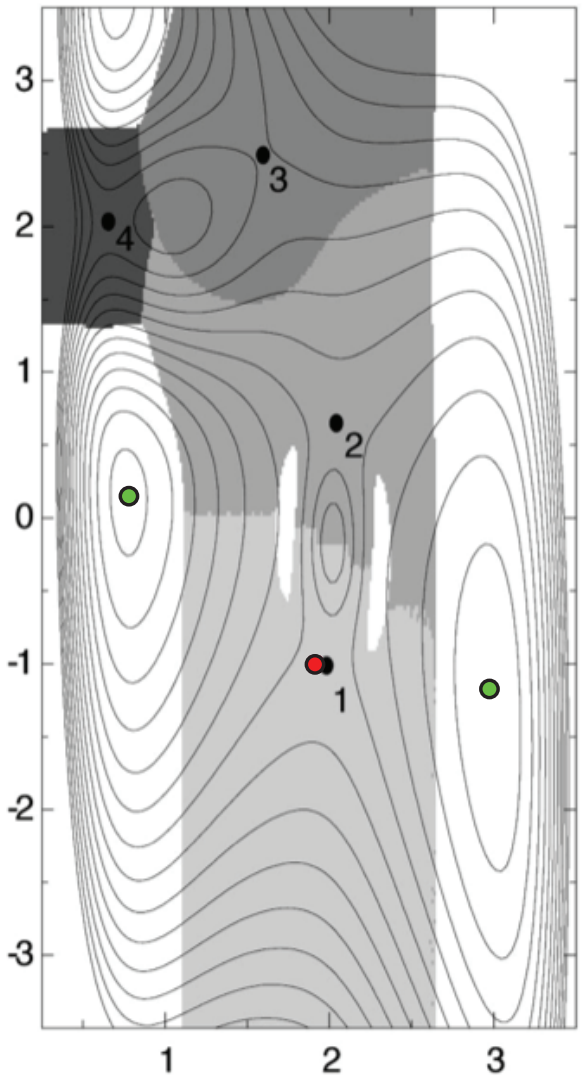


Using High Temperature Molecular Dynamics to Find Saddles

Replace random displacements with MD displacements.

Algorithm

1. Run high temperature MD
2. Periodically minimize the system to see if it has exited
3. Connect the initial minimum to the product minimum via a chain-of-states method such as Nudged Elastic Band
4. Find first maxima along the minimized band



Estimating the Error in the Rate

$$C = \frac{R(t)}{R(\infty)} \approx \frac{\langle R(t) \rangle}{R(\infty)}$$

$R(t)$: random variable of the total rate found after t seconds

Estimating the Error in the Rate

$$C = \frac{R(t)}{R(\infty)} \approx \frac{\langle R(t) \rangle}{R(\infty)} = \frac{\sum_{i=1}^{N} r_i p_i(t)}{\sum_{i=1}^{N} r_i}$$

$R(t)$: random variable of the total rate found after t seconds

N: total number of events

$p_i(t) = 1 - \exp(-t r_i^{\mathrm{Hot}})$: probability that event i has been found after t seconds of high temperature MD

r_i^{Hot}: can be calculated with HTST

Estimating the Error in the Rate

$$C = \frac{R(t)}{R(\infty)} \approx \frac{\langle R(t) \rangle}{R(\infty)} = \frac{\sum_{i=1}^{N} r_i p_i(t)}{\sum_{i=1}^{N} r_i} \approx \frac{\sum_{i \in F} r_i p_i(t)}{\sum_{i \in F} r_i}$$

$R(t)$: random variable of the total rate found after t seconds

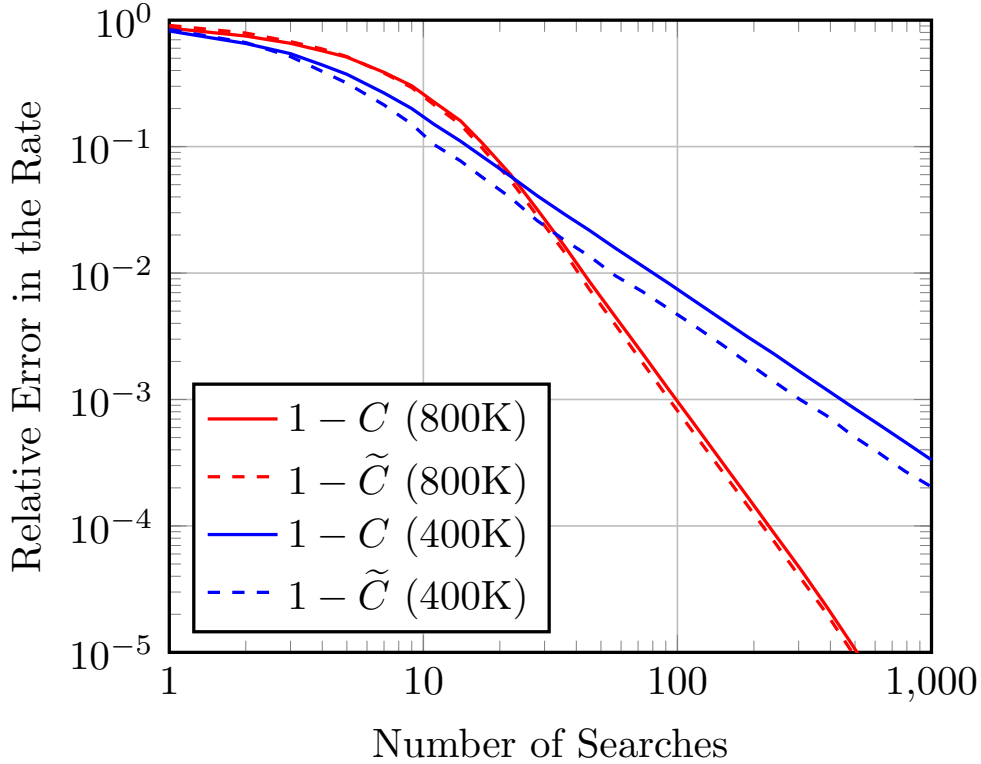
N: total number of events

$p_i(t) = 1 - \exp(-t r_i^{\mathrm{Hot}})$: probability that event i has been found after t seconds of high temperature MD

r_i^{Hot}: can be calculated with HTST

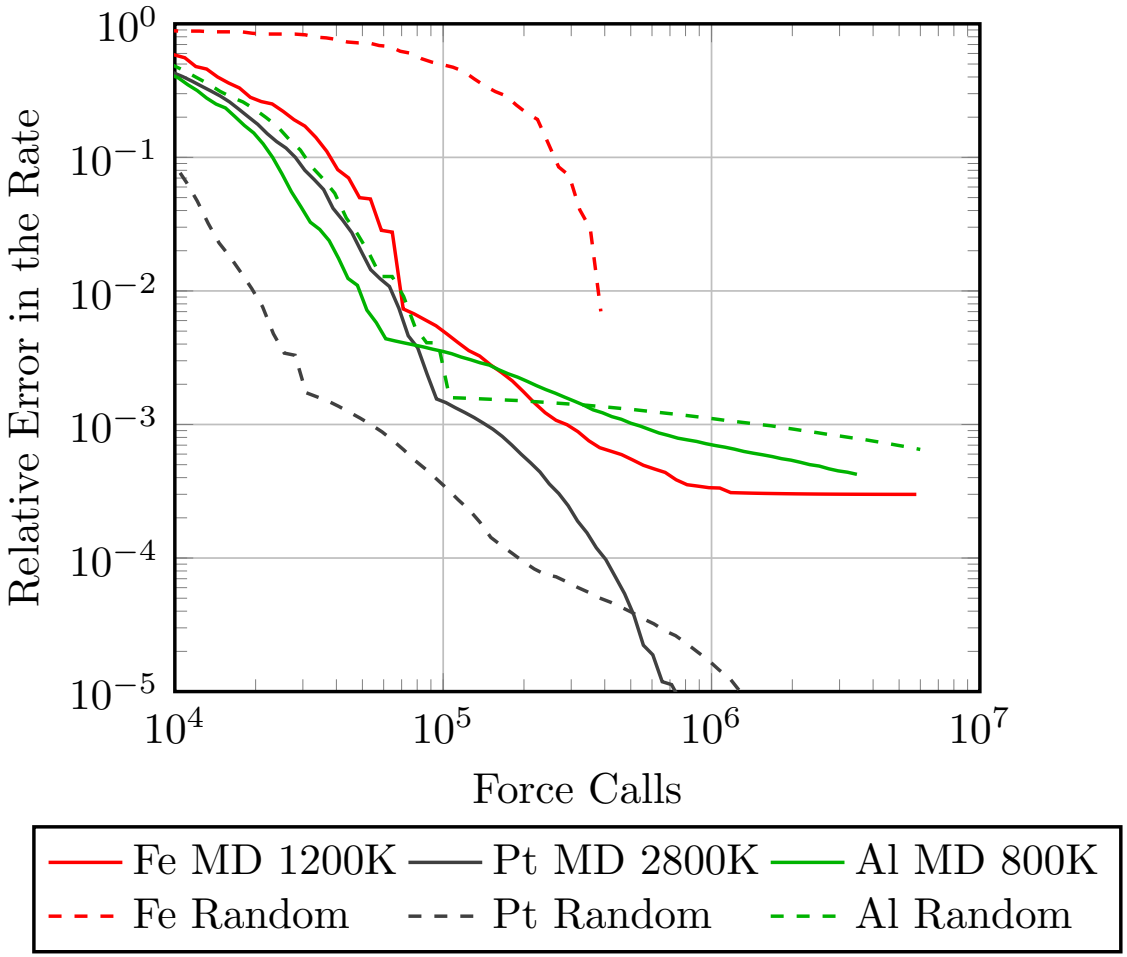
F: set of events that have been found

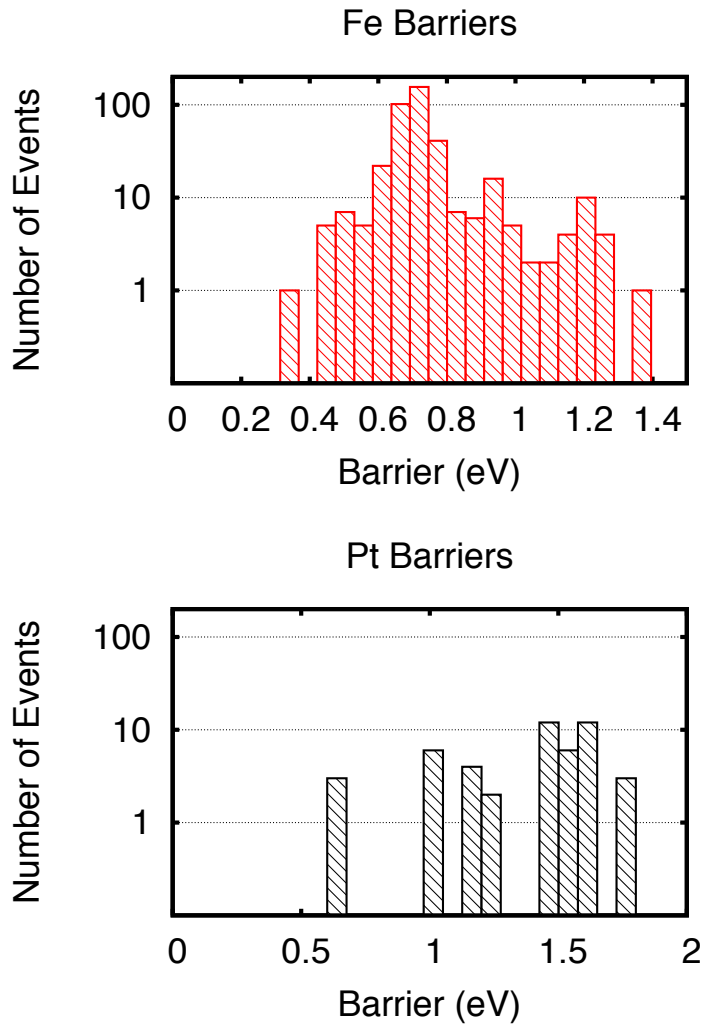
Evaluating The Quality of the Error Estimator



Times drawn from a harmonic system with rates calculated via HTST with 20 barriers linearly spaced between 0.1 and 0.4 eV.
Higher temperature sampling converges the total rate faster and reduces the bias of the estimator.

Saddle Search Method Comparison





Pros and Cons of MD Displacements compared to Random

Pros

- Finds events with probability proportional to their rate
- Can estimate a confidence in the total rate found
- Reduced chance of finding disconnected events
- No prior knowledge needed about reaction coordinate

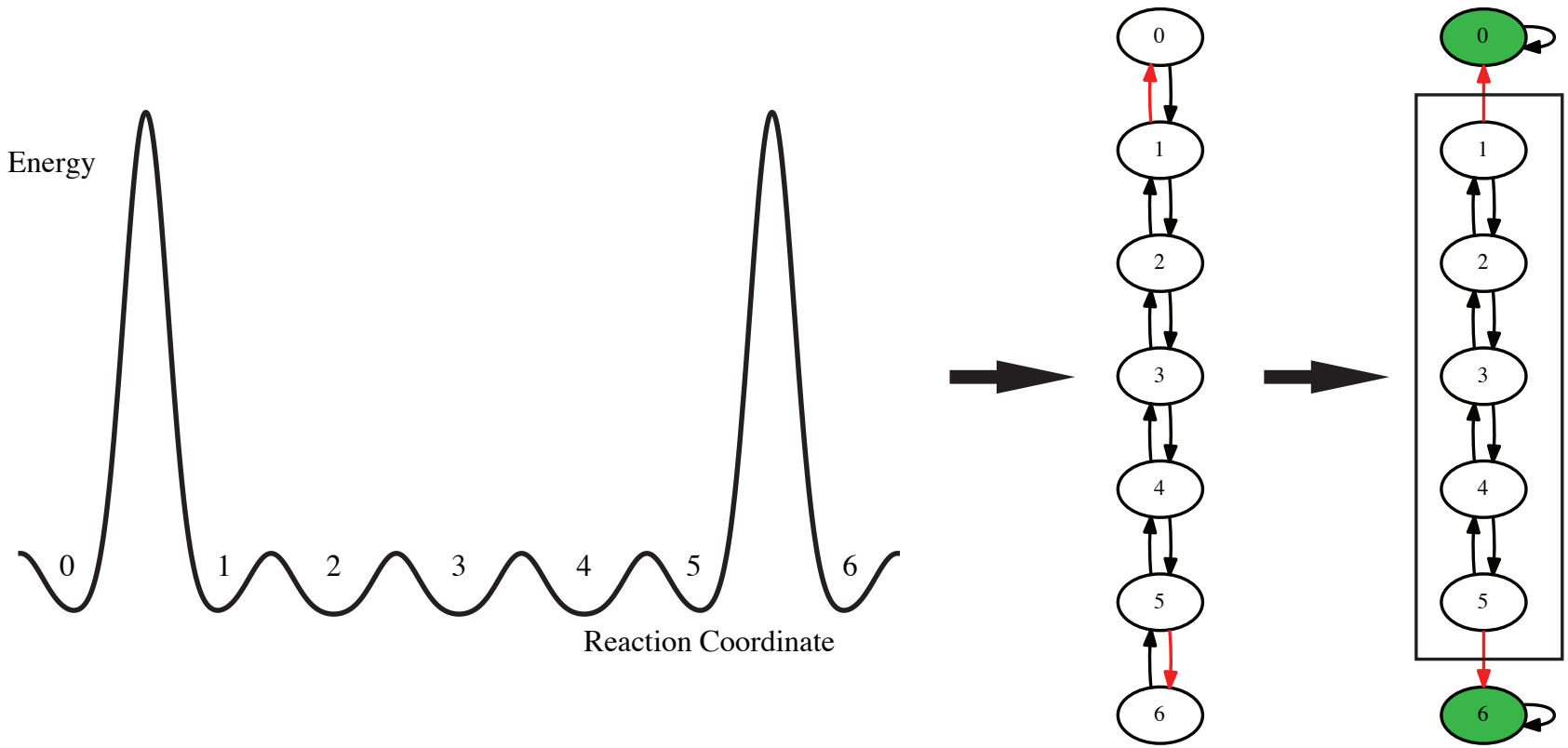
Cons

- MD displacement is more expensive than random
- NEB not guaranteed to find all pathways

Part II A second class of rare event problems

Superbasin

Superbasins are features of the potential energy surface, where there exist reactions that occur on widely differing timescales.



Monte Carlo with Absorbing Markov Chains

The *absorbing states*, r, are the exits from the superbasin and the *transient states*, s, are within the superbasin.

$$\mathbf{M}_{(r+s)\times(r+s)} = \begin{pmatrix} \mathbf{T}_{s\times s} & \mathbf{R}_{s\times r} \\ \mathbf{0}_{r\times s} & \mathbf{I}_{r\times r} \end{pmatrix}$$

Fundamental Matrix: mean number of times to visit state j starting at state i

$$\mathbf{N} = \sum_{k=0}^{\infty} \mathbf{T}^k = (\mathbf{I} - \mathbf{T})^{-1}$$

Matrix of absorption probabilities:

$$\mathbf{B} = \mathbf{NR}$$

$\mathbf{B}_{ij}$ is the probability of ending in state j when starting in state i

Mean time until absorption:

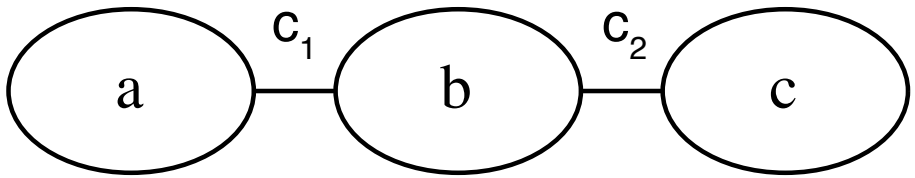
$$\mathbf{t} = \mathbf{N}\tau$$

τ is a vector of mean escape times from the transient states

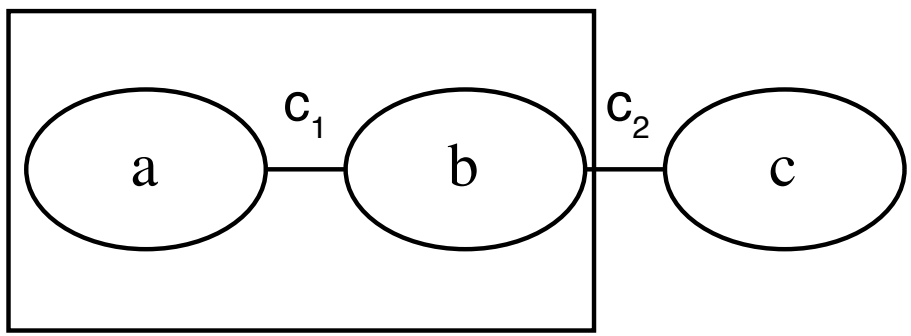
Grouping States During a KMC Simulation

Transition Counting

Identifying superbasins based on the number of times a transition has occurred.

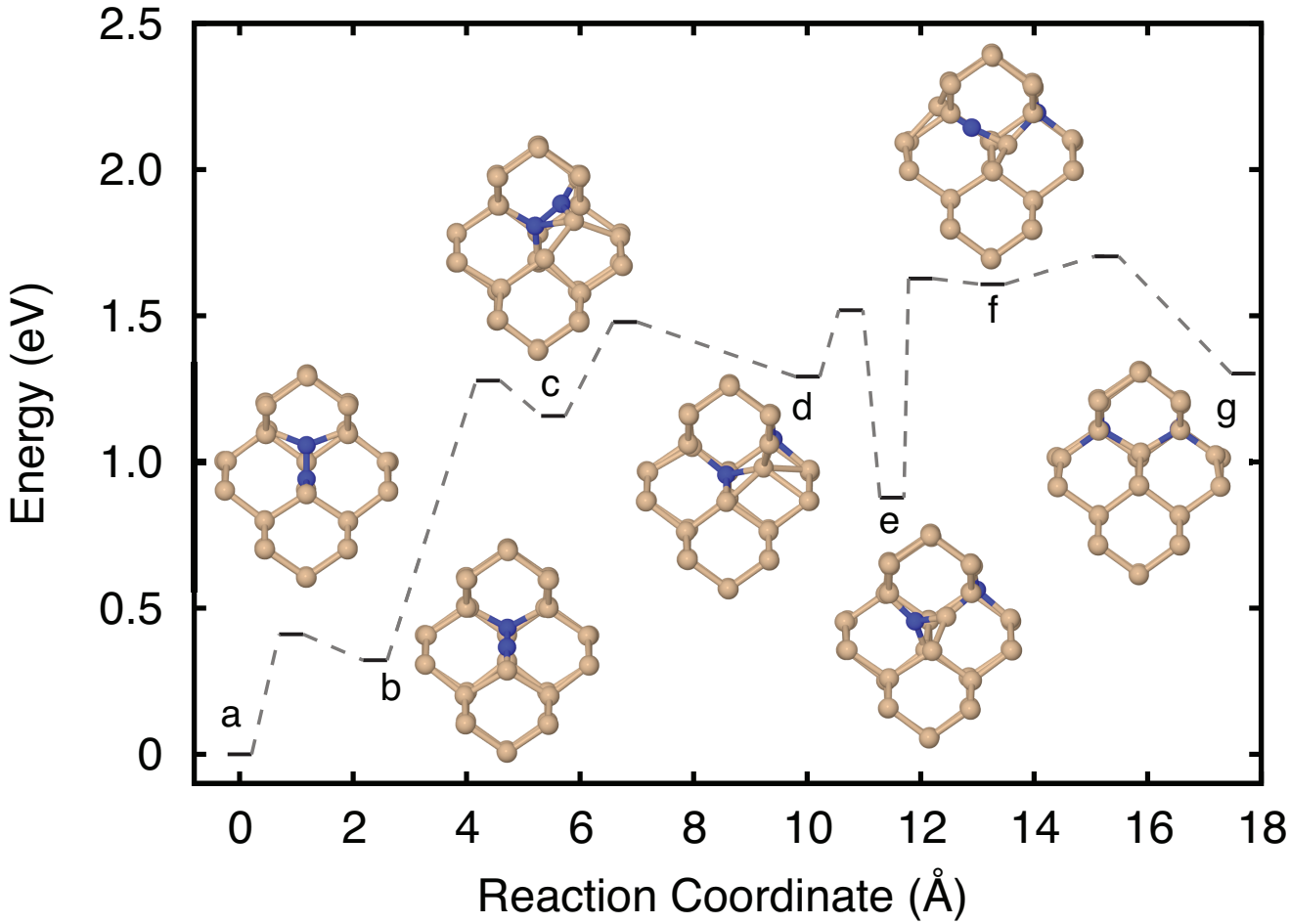


Increment the counter c_i each time a processes is followed. When $c_i \geq c_{\max}$ group the two states together.



Modeling SiB_2 cluster break-up with AKMC and DFT

States a and b form a superbasin that takes 5 billion steps to escape from on average at 300 K (about 20 days of computer time).



Performance of MCAMC

When modeling events on disparate timescales, extended precision types are sometimes needed.

N	`float` (7)	`double` (16)	`dd` (32)	`qd` (64)	`arb.` (154)
10	0.0004	0.0002	0.0004	0.0016	0.0083
100	0.0009	0.0011	0.0181	0.1768	1.0971
200	0.0025	0.0038	0.0932	0.9876	5.8999
500	0.0301	0.0439	1.1355	11.9715	64.7225
800	0.0602	0.1065	3.6915	39.6592	247.9850

Wall clock time in seconds for solving an absorbing Markov chain with N transient states. Computational effort is dominated by LU decomposition.

Drunkard's Walk

Open-source code for solving AMC problems in high precision.
`https://github.com/SamChill/drunkardswalk`

Eon: Open-source software for long timescale dynamics

Eon is software package that implements serveral long timescale dynamics methods. Details on obtaining the code at the end of this talk.

Methods

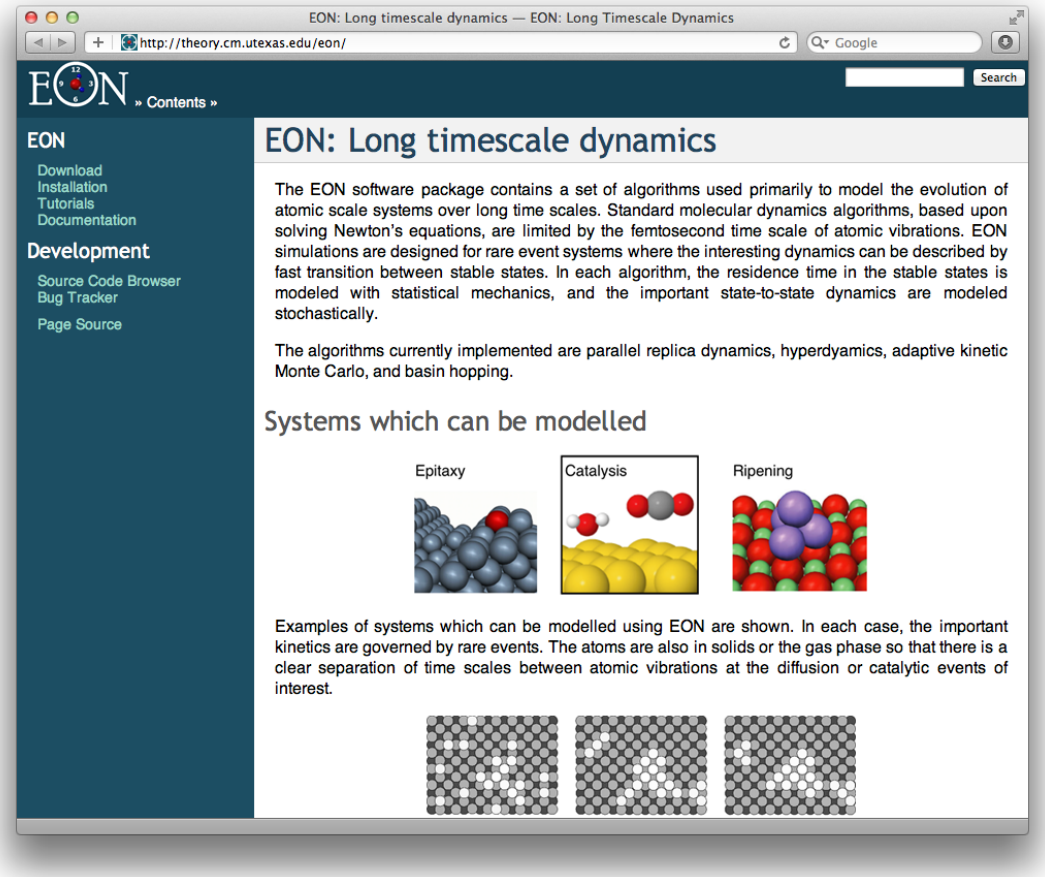
- Adaptive Kinetic Monte Carlo
- Parallel Replica
- Hyper Dynamics (Bond Boost)
- Basin Hopping
- κ-dynamics (in progress)
- Temperature Accelerated Dynamics (in progress)

Parallelization Options

- Local
- Job Queuing System (PBS, SGE, etc)
- BOINC (Distributed Computing)
- MPI

Potentials: EAM, LAMMPS, GPAW, VASP, and others.

Eon Homepage: http://theory.cm.utexas.edu/eon



S. T. Chill, M. Welborn, R. Terrell, L. Zhang, J. C. Berthet, A. Pedersen, H. Jonsson, and G. Henkelman *Model. Simul. Mater. Sci. Eng*, **Submitted 2013**.