# Quantifying the Performance of Kinetic Database in long timescale Dynamics Simulation

Completed for the Certificate in Scientific Computation

Date: 05/04/2020

Name: Mai Nguyen

Degree Program: Bachelor of Science in Chemistry (Computation)

**Supervisor: Wenrui Chai**
Department of Chemistry

**Signature:**

# Quantifying the Performance of Kinetic Database in long timescale Dynamics Simulation

Mai Nguyen

Department of Chemistry, University of Texas at Austin, Austin, Texas 78705, United States

**ABSTRACT:** Kinetic Database (KDB) has been used to speed up long timescale simulation in adaptive Kinetic Monte Carlo (aKMC) algorithm, an improved Kinetic Monte Carlo method, by providing a storage of saddle information. In order to evaluate the performance of Kinetic Database, two simple system, Pt heptamer island on Pt(111) surface and Al adatom diffusion on Al(100) surface, were tested in long timescale dynamics simulation. Three different saddle search conditions were analyzed: simulation with random searching (without KDB), simulation using KDB only and simulation using both random searching and KDB. With the concept of storing saddle information, the simulation involved KDB was expected to perform a less expensive work on finding saddle for aKMC simulation. To quantify the performance of a saddle search process, the number of force calls in saddle search as well as the number of saddles found was evaluated; a less expensive method would require less force calls in saddle search and find more saddles. Using statistic approach, several trials of simulation of similar initial configurations were also investigated. The result has shown that both Pt system and Al system statistically achieved higher saddle found in KDB than in random searching with the average force calls per saddle in random searching higher than KDB. Furthermore, comparing the force calls of random searching and KDB of a saddle in respect to its barrier has shows that the force call needed in saddle search of KDB could be decreased as the barrier gets lower.

## INTRODUCTION

The study of physical movements of atoms and molecules as well as the simulation of activated chemical reactions have been applied in many areas of theoretical chemistry. One common method, molecular dynamics, has been used in simulating systems specifically in respect to the timescale of atomic vibration at femtosecond.[1] However, the method becomes inefficient to more complicated systems when the time is tremendously large at an extensive scale, such as micro-second or longer. On that account, handling the convoluted systems with significant time requires the need of new methods that could improve the productivity of the simulation. Accordingly, methods called long timescale dynamic are introduced to simulate many complex systems such as battery and nanoparticle systems in material area. One of the methods that has been widely known is Kinetic Monte Carlo (KMC) using in atomic lattice.[2] Overtime, the method has been modified to overcome limitations of its own initial version called Adaptive Kinetic Monte Carlo (aKMC).

Kinetic Monte Carlo (KMC) simulates long timescale dynamic by using a random selection to move the system to a new state from a series of possible reactions determined at the beginning of the simulation.[3] This operation, however, gives rise to a few limitations. Without the list of possible reactions, the system cannot move forward to a new state explicitly; therefore, KMC restricts the system to only simple configurations. To handle a more complex system, the modified aKMC has been introduced without the requirements to know the possible reactions list in advance. The aKMC algorithm will search for possible reactions of the sytem for each simulation allowing the system to transit to any state without constraints such that the system can develop many convoluted configurations. Nonetheless, the searching for new reactions is exceptionally expensive, for the most part, when the configuration becomes more and more complicated.[4] In order to reduce the computational cost, a storage containing multiple sets of a saddle point and its reaction configurations is a good resolution from which aKMC can extract data to speed up the searching process.

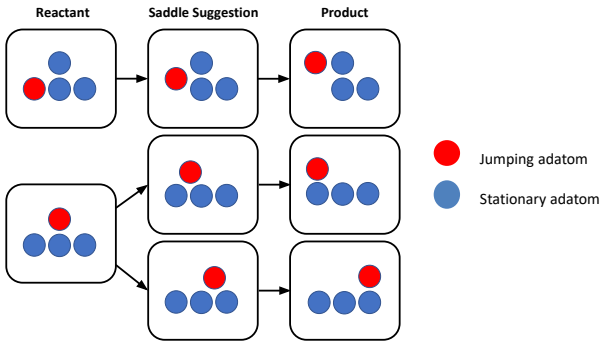



**Figure 1.** A storage of Kinetic Database (KDB) using mapping to suggest saddle point (middle column) as an output based on the reactant (left column) and the product (right column) as the input.

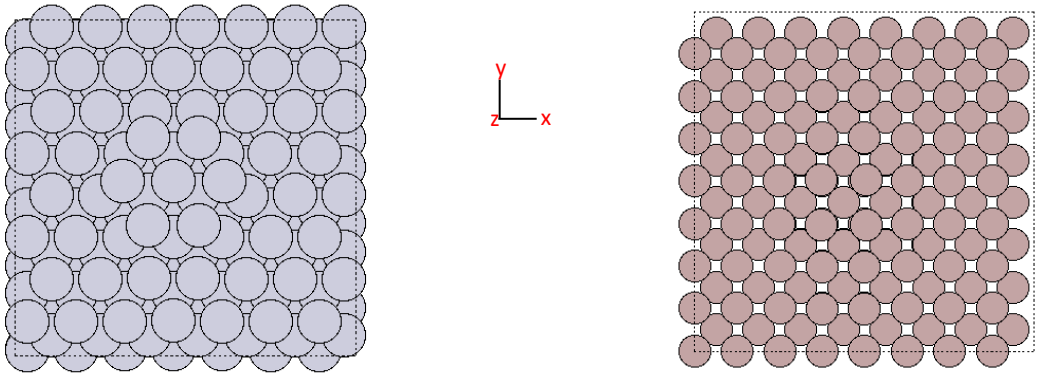
The storage used for aKMC, called a Kinetic Database (KDB), contain essential information for chemical reactions under a set of reactant, saddle point and product configurations of a certain state demonstrated in **Figure 1**. To help speeding up the searching process of aKMC, KDB will take the reactant and the product configurations as an input to apply the mapping method and suggest the saddle point configuration as an output.

In order to quantify the performance of KDB, different systems could be evaluated in simulating a long timescale dynamics where KDB is involved and where it is not. With the KDB's suggestion of saddle point from its database, the work of finding saddle point involving KDB was expected to be less computationally expensive. In other words, the suggestion of KDB was predicted to require less force calls in searching for saddles since it has stored a great deal of saddle points information in advance. Additionally, a hypothesis suggested that random searching will be expected to be more productive in searching saddle at lower barrier in comparison to KDB since saddles at lower barrier are easier for random searching to find.

The finding has verified that long timescale simulation involved KDB was less expensive through different tests statistically. Nonetheless, the hypothesis that suggested a better in productivity of random searching in saddle search was rejected by a contrasting result from the experiments. Instead, it has shown the evidence that the force call KDB uses for saddle search could be decreased as the barrier gets lower.

## MATERIALS AND METHODS

Calculations were performed with EON software package developed by the Henkelman and Jónsson research groups.[5] The software based upon Newton's method to simulate the evolution of atomic scale system over long timescale. Adaptive Kinetic Monte Carlo (aKMC) algorithm is used in EON. Two different system is investigated, Pt heptamer island on Pt(111) surface and Al adatom diffusion on Al(100) surface demonstrated in **Figure 2**.




a) Pt heptamer island on Pt(111)

b) Al adatom diffusion on Al(100)

**Figure 2.** a) Pt heptamer island on Pt(111) surface and b) Al adatom diffusion on Al(100) surface system

### 1. Pt heptamer island on Pt(111) surface system

The Pt heptamer island on Pt(111) surface system was simulated in Morse potential at 300 K. Geometry of the reactant was optimized before aKMC started searching for processes with converged force less than 0.001 eV/Å and less than 1000 iterations. Min-mode with Dimer method was used for saddle searching in this system.

The Dimer method searched for saddle point by using two images of the system with a difference in displacement of atoms.[6] The algorithm moved the dimer uphill toward the saddle point on potential energy surface and rotated the dimer during the movement to find its lowest curvature mode of the potential energy at the located point.[6] Atoms within 3.3 Å to the epicenter will be displaced for using Dimer algorithm.

A storage of processes was generated for the purpose of testing for KDB by simulating an arbitrary configuration of the system in a long simulation where the system evolved to thirty different states.

The system with all Pt monomers was investigated in a long simulation where the system evolved to twenty-one different states respectively with three different conditions: aKMC simulation with using random searching (without KDB), aKMC simulation with only KDB, and aKMC simulation with both KDB and random searching.

Six different configurations of the system were also investigated demonstrated in **Figure 3**: a) the Pt island was broken down to all monomers; b) the Pt island was broken down to one dimer and five monomers; c) the Pt island was broken down to one trimer and four monomers; d) the Pt island was broken down to one tetramer and three monomers; e) the Pt island was broken down to one pentamer and two monomers; f) the Pt island was broken down to one hexamer and one monomer. All atoms were positioned at hollow site for most productivity that would be validated later.

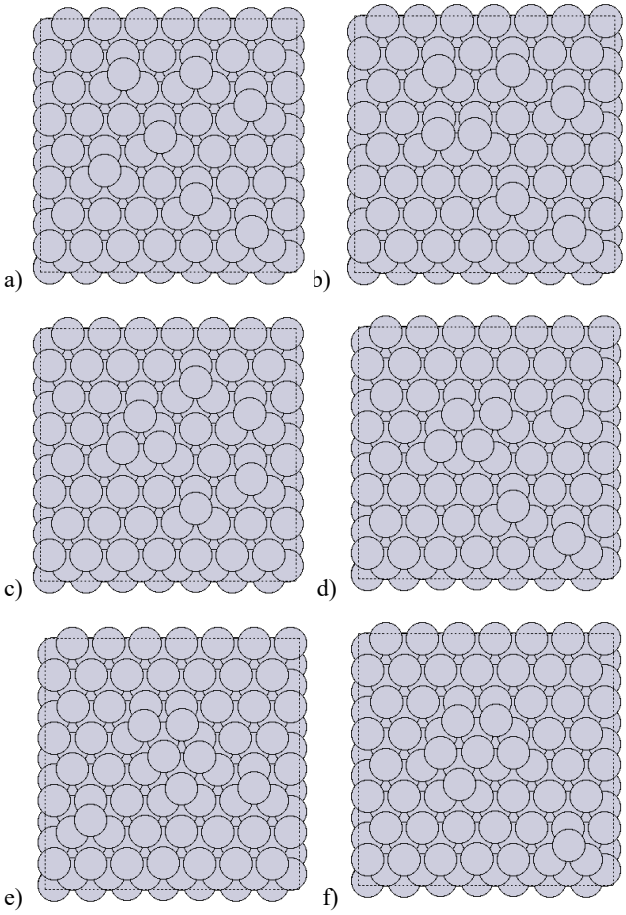



**Figure 3.** Six modified configurations of the Pt heptamer island on Pt(111) surface system.

Each configuration was reshaped in ten different structures for testing. EON will perform aKMC in each structure to investigate the saddle search process regarding one with KDB and one with random searching (without KDB).

### 2. Al adatom diffusion on Al(100) surface system

The Al adatom diffusion on Al(100) surface system was simulated with an embedded atom model developed by Voter and Chen.[7] Geometry of the reactant was optimized before aKMC started searching for processes with converged force less than 0.0001 eV/Å and less than 1000 iterations. Min-mode with Lanczos method was used for saddle searching in this system.

The Lanzcos method iteratively projected a vector on Halmiltonian, finding lowest eigenvalue and its eigenvector.[8] The convergence criteria for relative change in the estimated lowest eigenvalue is 0.05 eV/Å. Atoms within 5.5 Å to the epicenter will be displaced for using Lanzcos algorithm.

A storage of processes was generated for the purpose of testing for KDB by simulating an arbitrary configuration of the system in a long simulation where the system evolved to twenty states.

Two different configurations of the system were investigated demonstrated in **Figure 4**: a) the Al adatom was placed in hollow site and b) the Al adatom was placed in bridge site. The experiment also validated the productivity of finding saddle when atoms were at hollow site in comparison to bridge site.

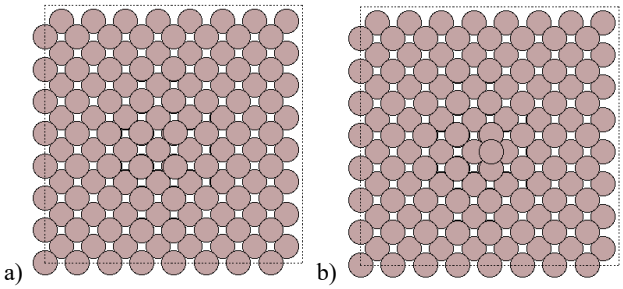



**Figure 4.** Two diffusion site of Al adatom on Al(100) surface system: a) hollow site; b) bridge site.

Each configuration was reshaped in ten different structures for testing. EON will perform aKMC in each structure to investigate the saddle search process regarding one with KDB and one with random searching (without KDB).

## RESULTS

### 1. Pt heptamer island on Pt(111) surface system

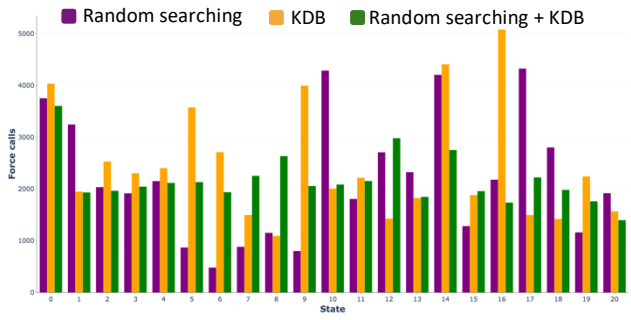



**Figure 5.** Total force calls for saddle search for three conditions: aKMC simulation with using random searching (without KDB), aKMC simulation with only KDB, and aKMC simulation with both KDB and random searching.

Figure 5 compares the total of force calls for saddle search in three different saddle search conditions: simulation with random searching (RS), with KDB only and with both random searching and KDB (RS + KDB). The results do not have enough evidence to support which saddle search condition uses more force calls than the others since the figure shows inconsistent trend for each state. To produce a decisive outcome, the result from Figure 5 was taken into account with the number of saddles found for each state. Figure 6 demonstrates the average force calls per saddle found computed from the previous result for three different saddle search conditions: RS, KDB only and RS + KDB. The new result then shows that the simulation with RS achieved the highest average force calls per saddle for most states.

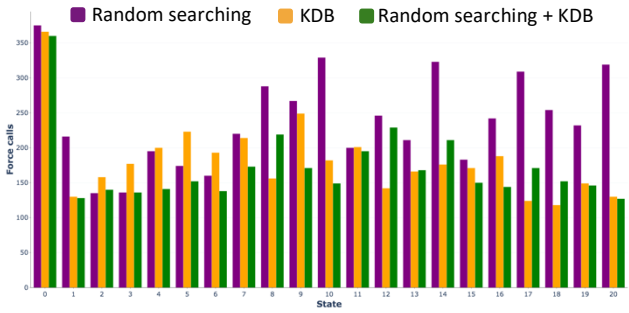



**Figure 6.** Average force calls per saddle for three conditions: aKMC simulation with using random searching (without KDB), aKMC simulation with only KDB, and aKMC simulation with both KDB and random searching.

KDB and RS + KDB was respectively compared to RS by taking a ratio of average force calls per saddle. The results in Table 1 shows that the ratio of KDB to RS has an average of 0.814 ± 0.065 with 95% confidence interval 0.814 ± 0.030; the ratio of RS + KDB to RS has an average of 0.745 ± 0.041 with 9% confidence interval 0.745 ± 0.019. It has shown that KDB and RS + KDB respectively have used a number of force calls per saddle approximately 20% less than RS method.

**Table 1.** Force call usage comparison between using KDB and RS + KDB to RS method.

| Method | Average | St. D. | RSD | 95% Conf. Interval |
|---|---|---|---|---|
| KDB | 0.814 | ± 0.065 | 0.080 | 0.814 ± 0.030 |
| RS + KDB | 0.745 | ± 0.041 | 0.055 | 0.745 ± 0.019 |

Six configurations in Figure 3 of the Pt system were evaluated in ten different structures (or states) respectively. Figure 7 shows the comparison in number of saddles found and average force call per saddle between RS and KDB simulation for the first configuration: Pt island all broken down to monomers at hollow-site. The average force call per saddle developed a trend showing RS has a higher average force calls for most states.

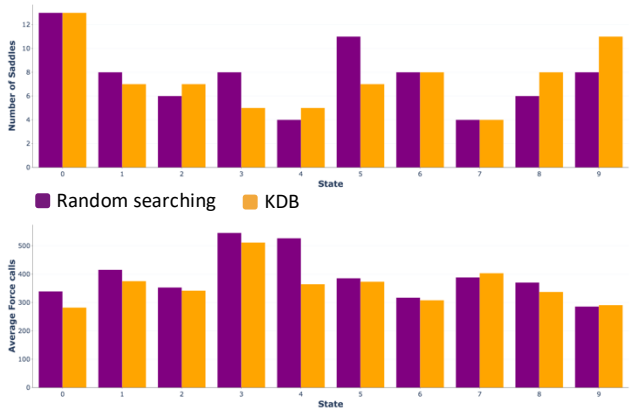



**Figure 7.** Comparison of number of saddles found and average force calls per saddle for the configuration of all monomers Pt between random searching and KDB simulation.

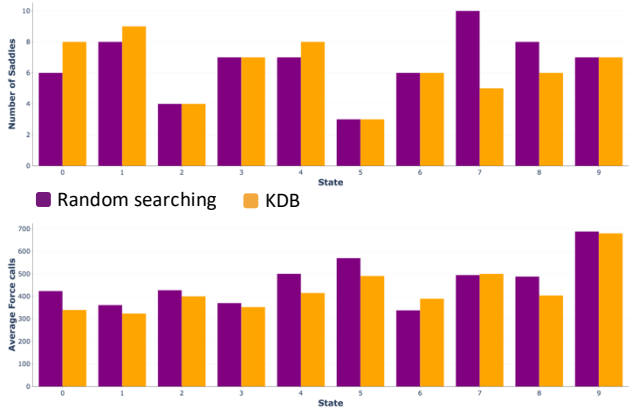



**Figure 8.** Comparison of number of saddles found and average force calls per saddle for the configuration of one dimer Pt between random searching and KDB simulation.

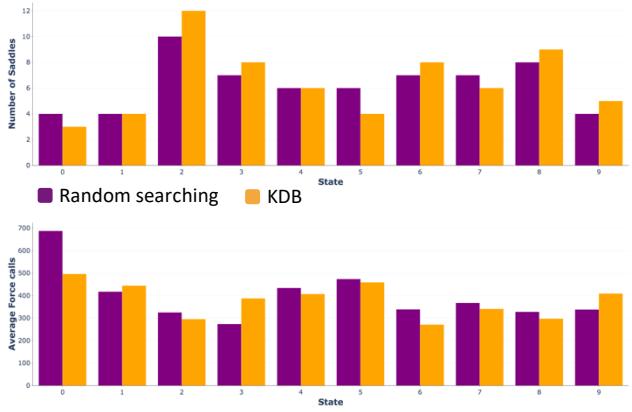



**Figure 9.** Comparison of number of saddles found and average force calls per saddle for the configuration of one trimer Pt between random searching and KDB simulation.

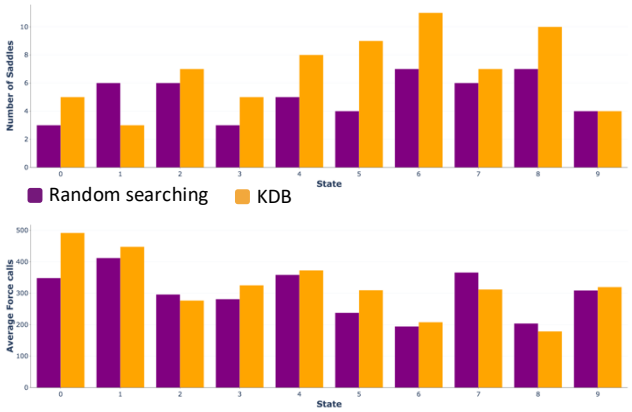



**Figure 10.** Comparison of number of saddles found and average force calls per saddle for the configuration of one tetramer Pt between random searching and KDB simulation.

Similarly, Figure 8, Figure 9 and Figure 11shows similar trend in both analysis for the configuration where there was a dimer and a trimer and a pentamer of Pt respectively. Figure 10 with the configuration of a Pt tetramer exceptionally indicates that KDB developed higher average force calls per saddle.

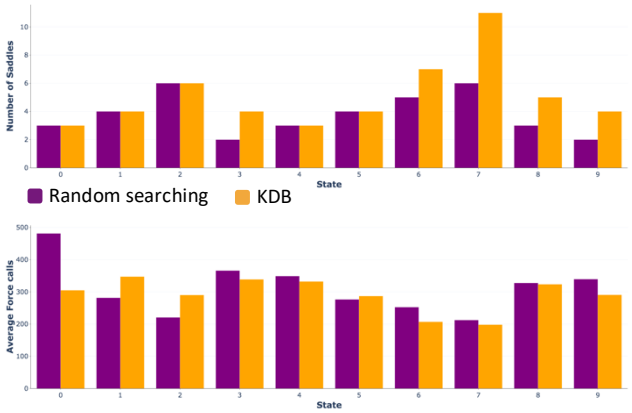



**Figure 11.** Comparison of number of saddles found and average force calls per saddle for the configuration of one pentamer Pt between random searching and KDB simulation.

Figure 12 shows a discrepancy in number of saddles found between KDB and RS. The result indicates that the configuration of a Pt hexamer was not preferable for RS to find saddle in comparison to KDB. In addition, the trend with higher average force calls per saddle in RS was also not shown in this configuration.

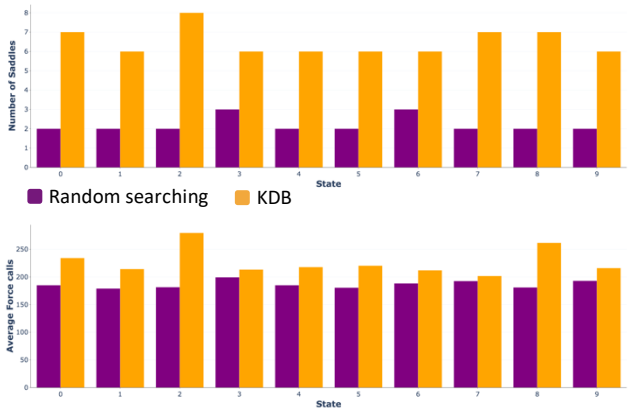



**Figure 12.** Comparison of number of saddles found and average force calls per saddle for the configuration of one hexamer Pt between random searching and KDB simulation.

**Table 2.** Force call usage comparison between using KDB to RS method in different configurations.

| Configuration | Average | St. D. | RSD | 95% Conf. Interval |
|---|---|---|---|---|
| All monomers | 0.924 | ± 0.033 | 0.036 | 0.924 ± 0.022 |
| One dimer | 0.926 | ± 0.036 | 0.039 | 0.926 ± 0.024 |
| One trimer | 0.986 | ± 0.067 | 0.068 | 0.986 ± 0.044 |
| One tetramer | 1.077 | ± 0.059 | 0.055 | 1.077 ± 0.039 |
| One pentamer | 0.970 | ± 0.065 | 0.067 | 0.970 ± 0.043 |
| One hexamer | 1.221 | ± 0.053 | 0.043 | 1.221 ± 0.035 |

The ratio of force calls per saddle between KDB and RS method in Table 2 shows that KDB have used force calls per saddle approximately 5% less than RS in most of the configurations.

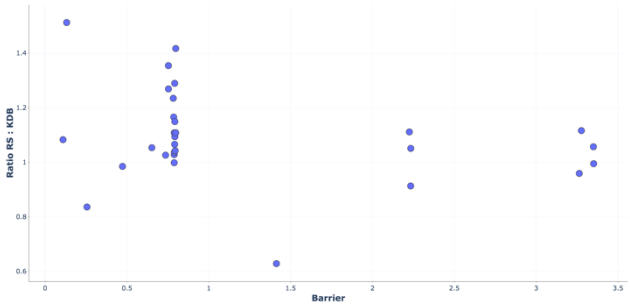



**Figure 13.** Distribution of ratio between random searching and KDB force calls for a similar saddle in Pt system and its barrier.

In order to compare the force calls of a certain saddle, all the saddles of 60 states from six configurations were analyzed and selected to only the saddles with the same barrier. After that, the ratio between the force calls of RS and KDB for each of the saddle was computed versus its barrier demonstrated in Figure 13. All saddle less than 0.1 eV were disregarded. The distribution indicates RS and KDB used similar force calls to find saddle at higher barrier while RS used more force calls in comparison to KDB as the barrier of the saddle got lower.

### 2. Al adatom diffusion on Al(100) surface system

Two configurations of Al system in Figure 4 were evaluated in ten different structures (or states) respectively. Figure 14 shows the result of the configuration which Al adatom locating at hollow-site. KDB explicitly surpassed RS and found more than half of the saddles found in RS. Also, RS developed much higher average force calls per saddle in comparison to KDB for most states. Figure 15 indicates that the configuration with Al adatom at bridge-site shared a similar trend for number of saddles found. However, the average force calls per saddle in KDB was higher.

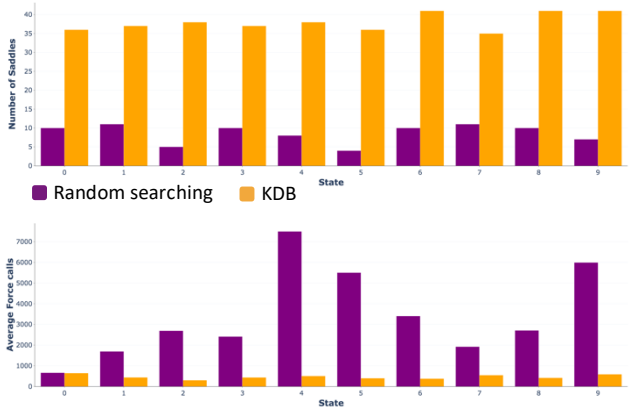



**Figure 14.** Comparison of number of saddles found and average force calls per saddle for the configuration of hollow-site Al adatom between random searching and KDB simulation.

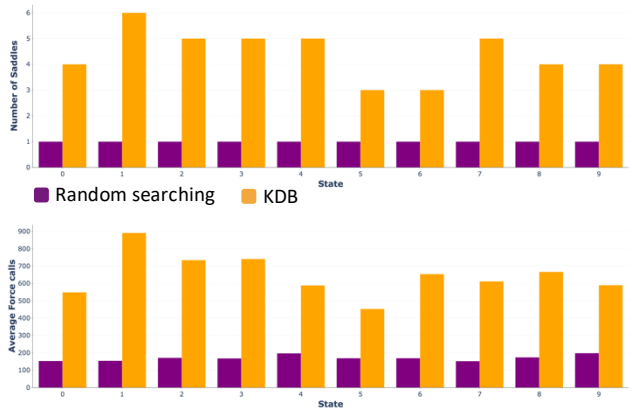



**Figure 15.** Comparison of number of saddles found and average force calls per saddle for the configuration of hollow-site Al adatom between random searching and KDB simulation.

**Table 3.** Force call usage comparison between using KDB to RS method in different binding site.

| Binding Site | Average | St. D. | RSD | 95% Conf. Interval |
|---|---|---|---|---|
| Hollow | 0.231 | ± 0.090 | 0.390 | 0.231 ± 0.060 |
| Bridge | 3.847 | ± 0.299 | 0.078 | 3.847 ± 0.197 |

In hollow-site Al atom configuration, the ratio of average force calls per saddle between KDB and RS is lower than 1 indicating KDB performs saddle search more efficiently. On the other hand, the high ratio of average force calls per saddle between KDB and RS in bridge-site Al atom configuration demonstrates that KDB requires much higher force calls to find saddles.

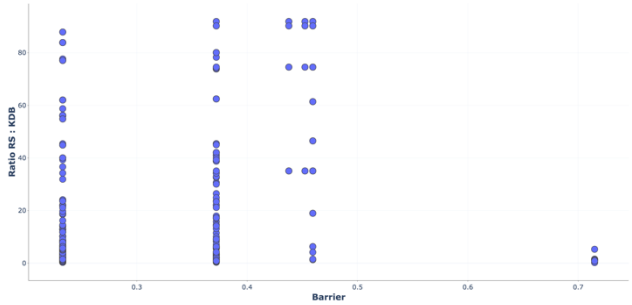



**Figure 16.** Distribution of ratio between random searching and KDB force calls for a similar saddle in Al system and its barrier.

Figure 16 demonstrates the ratio between random searching and KDB force calls versus its barrier. The result shows a lower ratio at higher barrier similarly to that of the Pt system. Nevertheless, the ratio at lower barrier does not produce a decisive conclusion.

## DISCUSSION

The comparison in the total of force calls between three saddle search condition in the Pt system was inconclusive due to sampling error. Each state for each condition was respectively different from each other; as a result, some state will have exceptionally complicated configuration than the other which leads to less force calls used. On top of that, the result does not provide any conclusive trend which confirms the poor approach. To make the result more applicable, taking the average force call per saddle improves the problem. The improved result has shown that the average force calls per saddle in RS was higher for most of the states. Table 1 has also shown that the ratio of average force calls per saddle of KDB and RS + KDB to RS method were both less than 1 indicating fewer force calls needed for saddle search. The relative uncertainty is less than 10% determining the validity of the test. Nonetheless, comparing the average force calls per saddle between methods while each configuration not matching for each state might not produce the best conclusion statistically.

For that reason, testing similar configurations to compare the force calls of each method is definitely a more desirable strategy to draw the expected conclusion. The Pt heptamer island on Pt(111) surface system is considered one of the common systems that has been used for testing long timescale dynamics for aKMC algorithms. Due to its simplicity, 60 trials have been used to test the hypothesis statistically. Indeed, more convoluted systems will significantly require more assessments in order to evaluate the performance of a method. As a result, the result shows that KDB achieved an equal or higher in number of saddles found than RS for most cases. In respect of the average force call per saddle, Table 2 indicates that KDB developed a lower average force calls per saddle. The statistical analysis from Table 2 also indicates that the relative uncertainties are less than 7% which can validate the accuracy of the test.

Al adatom diffusion on Al(100) surface system is also a simple system that could be tested with small test size. The result in the configuration of Al adatom located at bridge site and hollow site shared similar trends in the number of saddles found for each method. Nonetheless, Table 3 shows that KDB works exceptionally more efficient in finding saddles in comparison to RS in hollow site but works significantly less efficient in bridge site. The comparison between Al adatom at hollow-site and bridge-site could provide the insight about which binding site is more productive in evaluating the saddle search process. The results from Figure 14 and 15 show that hollow-site produced a better performance of KDB in comparing to the bridge-site. This observation could demonstrate that binding site might affect KDB's performance to find saddle within the existence of a saddle storage. To further evaluation of KDB use in long timescale simulation, hollow-site would be the recommended binding site for testing system.

In comparison between the RS force calls and the KDB force calls for a certain saddle point in two systems, RS used more force calls than KDB at lower barrier saddle, while approaching approximately similar force calls to KDB as the barrier went up. There was no insight on how to explain the current observed trend. This observation, instead, can support the hypothesis about the efficiency in using KDB for searching saddle. Since the lower barrier saddles are easier to find, the probability of these saddles has been stored in KDB is higher. For this reason, KDB should use less force calls to find these saddles. As the barrier goes up, the probability of these saddles inserted in KDB gets lower as these saddles are harder to find. On that account, KDB might spend more work to look for these saddles to the stage where it has to spend equal amount of work to RS in order to find these saddles. The result has shown that saddles at lower barrier are indeed easier to

find but it does not necessarily mean that RS could find it more productively than KDB.

The experiments quantifying the performance of KDB in long timescale dynamics simulation have given credence to the efficiency of KDB in the simulation. However, the accuracy of KDB could be indirectly interpreted by using a great number of similar tests as KDB was observed in a few tests to find a greater number of saddles comparing to using RS.

## ASSOCIATED CONTENT

### Computer Code

Sample programming codes on computation and calculation for Pt system are included in Appendix.
All programming codes on computation and calculation for the projects are included in Github repository given in the following link. Instructions and guidance about the programming codes are recorded in README.txt file.
https://github.com/maiihn/SDS379R_Spring2020

## ACKNOWLEDGMENT

I thank Dr. Juliana Duncan and Dr. Wenrui Chai as the supervisor of the project; I also thank Naman Katyal for giving instructions of the calculation. I also thank Dr. Graeme Henkelman for providing platform for computation. Funding, instruments and software were provided by the Henkelman group and the University of Texas at Austin.

## LITERATURE CITED

(1) Henkelman, G. Annual Review of Materials Research, **2017**, 47, 199–216 .
(2) Mason, D.R.; Hudson, T.S.; Sutton, A.P. *Computer Physics Communications*, **2005**, 165(1): 37–48.
(3) Henkelman, G.; Xu, L. *The Journal of Chemical Physics*, **2008**, 129, 114104.
(4) Terrell, R.; Wellborn, M.; Chill S.T.; Henkelman, G. *The Journal of Chemical Physics*, **2012**, 127, 014105.
(5) Theory.cm.utexas.edu. n.d. *EON: Long Timescale Dynamics — EON: Long Timescale Dynamics*. [online] Available at: <http://theory.cm.utexas.edu/eon/index.html#> [Accessed 28 April 2020].
(6) Henkelman, G.; Jónsson, H. *J. Chem. Phys.,* **1999**, 111(15), 7010–7022.
(7) Chill, S.T.; Henkelman, G. *J. Chem. Phys.,* **2014**, 140, 214110.
(8) Malek, R.; Mousseau, N. *Physical Review*, **2000**, 62(6), 7723–7728.

**APPENDIX**

## 1. Extraction Process

Environment: Terminal

Server: fri.oden.utexas.edu using ssh

**fc_export.py** used in three saddle-search condition: This script extracts information about the number of force calls for each state and its number of saddles found.

```
#!/usr/bin/env python3

import os
import numpy as np
import export
from sys import argv

if len(argv) != 2 or '-h' in argv:
    print('Use flag \n \t \t -rs    :  random searching \n \t \t -kdb   :  only kdb \n \t \t -both  :  kdb
+ random searching')
    exit()
elif len(argv) == 2:
    if argv[1] not in ['-rs','-kdb','-both']:
        print('Invalid method')
        print('Use flag \n \t \t -rs    :  random searching \n \t \t -kdb   :  only kdb \n \t \t -both  :
kdb + random searching')
        exit()
    else:
        method = argv[1]

path = export.Method('akmc-pt-')
pathway = path.get_method(method)

thepath = '/home/maiihn/code/eon/calculation/pt-system/' + pathway + '/states/'
thenewpath = '/home/maiihn/code/eon/calculation/pt-system/OUTPUT/' + pathway + '/force_call' + '/re-
sult_state'
num_state = export.Number(thepath).get_num_state() - 1

for i in range(num_state):
    state = i

    path = thepath + str(state)
    newpath = thenewpath + str(state)
    num_saddle = export.Number(path).get_num_saddle()

    doc = export.Doc(path)
    doc.export_forcecall(newpath,num_saddle)

# do the sums for force calls
path = thepath
newpath = '/home/maiihn/code/eon/calculation/pt-system/OUTPUT/' + pathway + '/force_call' + '/state_re-
sult'
doc = export.Doc(path)
doc.export_sum_forcecall(newpath,num_state)


print('The csv of the force calls in', num_state, 'states for', pathway, 'has been exported in the OUTPUT
directory.')
```

## 2. Analysis Process

Environment: Jupyter notebook

Server: Local

**State_saddle.ipynb** used in comparing KDB and RS in six different configurations: This script import the output from the extraction process to plot some statistical analysis.

```
import pandas as pd
import plotly as plt
import matplotlib as mplt
```

```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import numpy as np
from sqlalchemy import create_engine
engine = create_engine('sqlite://', echo=False)
from pandas.io import sql

rs = pd.read_csv('ransearch/state.csv')
kdb = pd.read_csv('onlykdb/state.csv')
tab = [rs,kdb]
name_list = ['Random Searching', 'KDB',]
color_list = ['purple','orange']

fig = make_subplots(rows = 2, cols = 1)

fig.update_layout(showlegend = False,
                  xaxis = dict(title_text = '<b>State<b>', title_font = {'size':40}),
                  xaxis2 = dict(title_text = '<b>State<b>', title_font = {'size':40}),
                  yaxis = dict(title_text = '<b>Number of Saddles<b>', title_font = {'size':40}),
                  yaxis2 = dict(title_text = '<b>Average Force calls<b>', title_font = {'size':40}),
                  xaxis_type = 'category',
                  xaxis2_type = 'category',
                  template = 'plotly_white',
                  width=3000, height=2000)
fig.update_xaxes(ticks='outside', showline=True, linewidth=1, linecolor='black', tickfont=dict(size=30))
fig.update_yaxes(ticks='outside', showline=True, linewidth=1, linecolor='black', tickfont=dict(size=30))

for i in range(len(tab)):
    fig.add_trace(go.Bar(x= tab[i]['state'],\
                         y = tab[i]['unique saddles'],
                         name = name_list[i],
                         marker_color = color_list[i]), row = 1, col = 1)


for i in range(len(tab)):
    fig.add_trace(go.Bar(x=tab[i]['state'],
                         y=tab[i]['force call saddle']/tab[i]['unique saddles'],
                         name = name_list[i],
                         marker_color = color_list[i]), row = 2, col = 1)
fig.show()
```

**REFLECTION**

Kinetic Database was developed by the Henkelman group with the purpose of speeding up the long timescale dynamics simulation using adaptive Kinetic Monte Carlo algorithm. The project has helped exploring the performance of Kinetic Database (KDB) in long timescale dynamics simulation. By applying the concept of a database, the development of KDB was expected to be useful in researches involving a large scale of dynamic simulations. By testing two simple models of Pt system and Al system, the results have shown a desirable aid on speeding up the most important part of simulating a system, finding the possible processes for the next state. The project also demonstrates that the development of KDB has meet the general purpose of generating a storage of processes to be reusable. For that reason, the result of the project could assist on the ongoing development of KDB for better improvement as well as better query algorithms.

Alongside with the help in chemistry insight, the project has helped in approaching and improving statistical analysis skills to examine a hypothesis. Challenges that have been encountered during the project is specifically choosing testing samples as well as the size of the samples. As a result, many unexpected results were achieved leading to many changes and modifications of testing samples. After that, using more testing samples as well as selecting a more compatible testing samples have given a more desirable result.

Furthermore, I have learned significantly soft skills as I have worked with two supervisors throughout this project. The two supervisors have helped me working through my projects efficiently. The only problem that I have faced was that the sudden change of the first supervisor to the second supervisor has given me a hard time to modify the changes in insight of the project on time. As a result, a lot of testing samples and testing systems were cut off in order to keep the time on track. Through this challenge, I believed I have developed more communication and problem-solving skills.

The result of this project has reflected a significant validation of the development of KDB in long timescale dynamics simulation. Furthermore, I believed that using the concept of this project at an extensive scale could eventually produce a more convincing evidence showing that KDB is more efficient and more accurate in finding processes in adaptive Kinetic Monte Carlo's simulating long timescale dynamics. The project can support further researches involving KDB for the means of performing adaptive Kinetic Monte Carlo simulation as well as support the development of KDB to a better improvement.