Screening Potential New Drugs with HTC

Open Science Grid User School, July 29, 2016

Spencer Ericksen

UW-CCC, Drug Discovery Core, Small Molecule Screening Facility

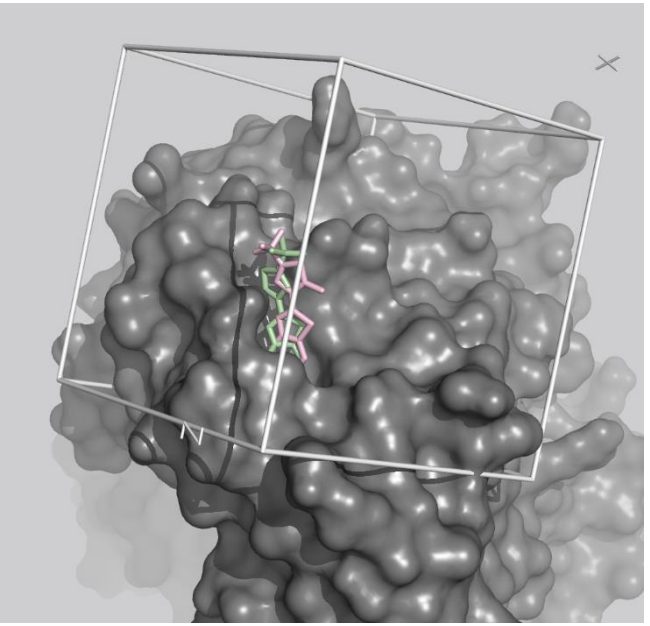
sericksen@wisc.edu

Carbone Cancer Center
UNIVERSITY OF WISCONSIN
SCHOOL OF MEDICINE AND PUBLIC HEALTH

CENTER FOR
HIGH THROUGHPUT
COMPUTING

Overview

- What is vHTS?
- What is docking?
- How expensive is docking?
- How do we scale on HTC resources?
- How do we benefit from HTC?
- What have we learned using HTC?

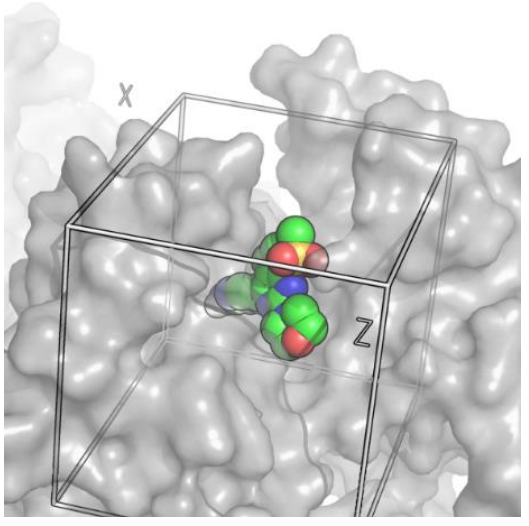


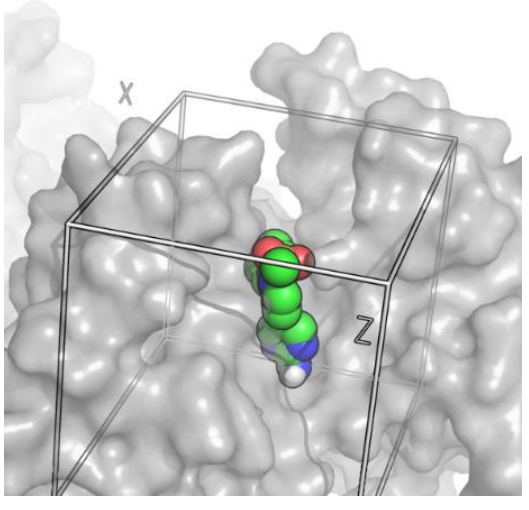
What is **v**HTS?

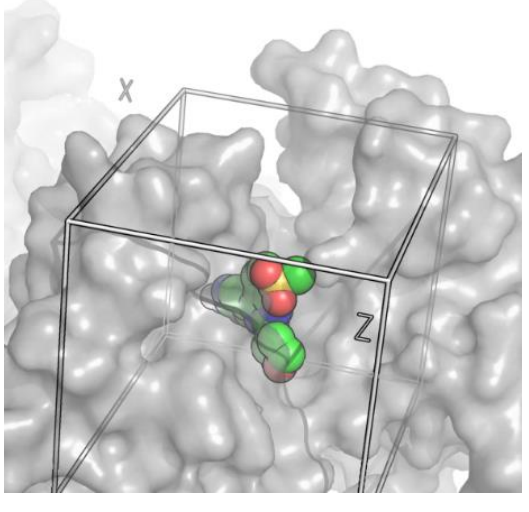
- Drug Discovery: looking for needle in haystack.
- **HTS** assays of 10,000s to millions compounds.
- $1 per compound!
- Filter using **vHTS** first! Prioritize a subset for screening.
- If vHTS can provide 100-fold enrichment in top 1%, reduce a 100,000 set to a 1,000 compound subset.
- Use **docking** to predict potential for compound-**target** interaction.

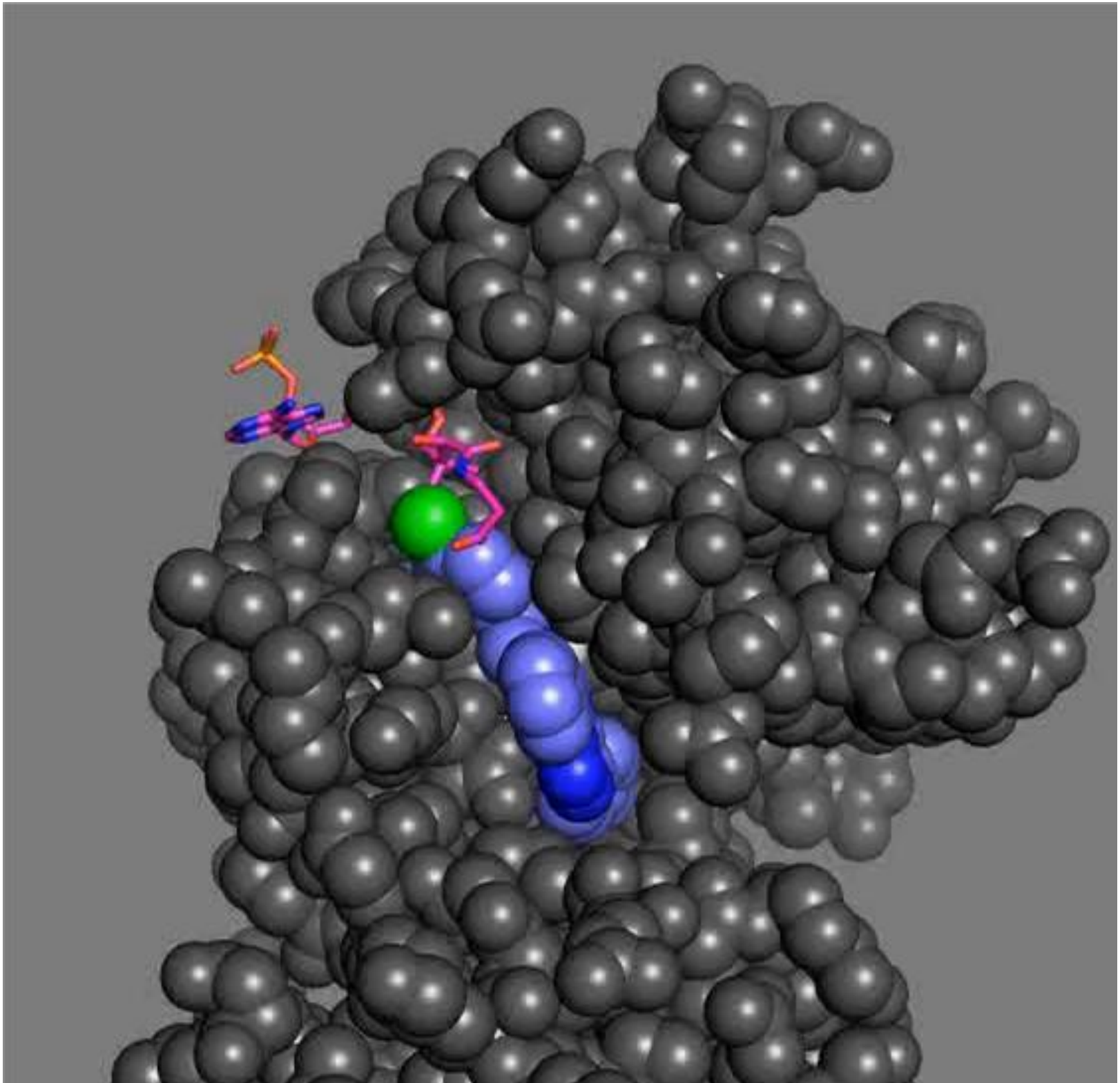
What is docking?

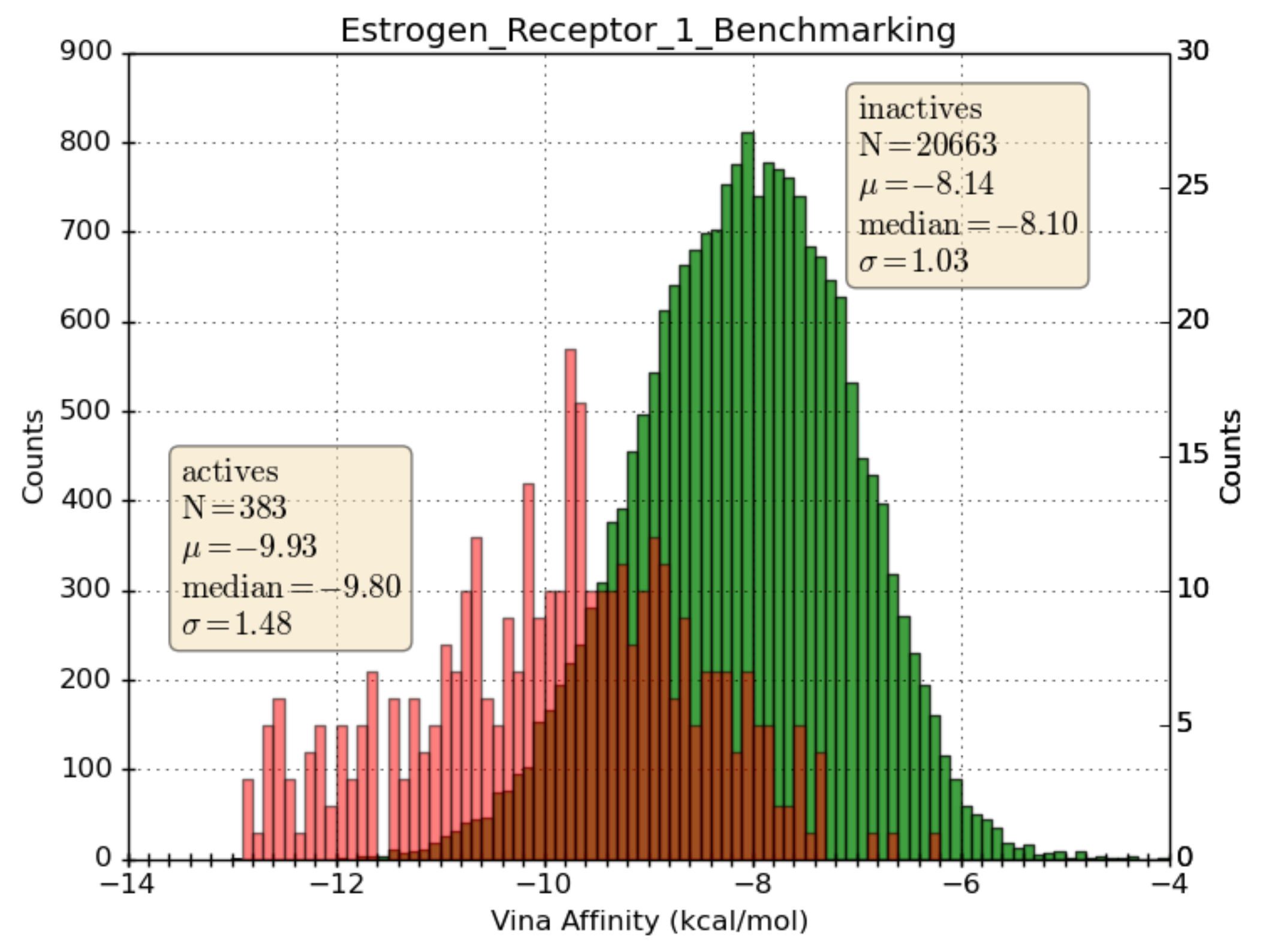
- Docking looks for best compound binding orientation on a target.
- Search is guided by a scoring function that evaluates favorability of each sampled configuration.
- Many docking programs exist with different search strategies and scoring functions.
- Docking score is crude estimate of binding favorability for a given compound.





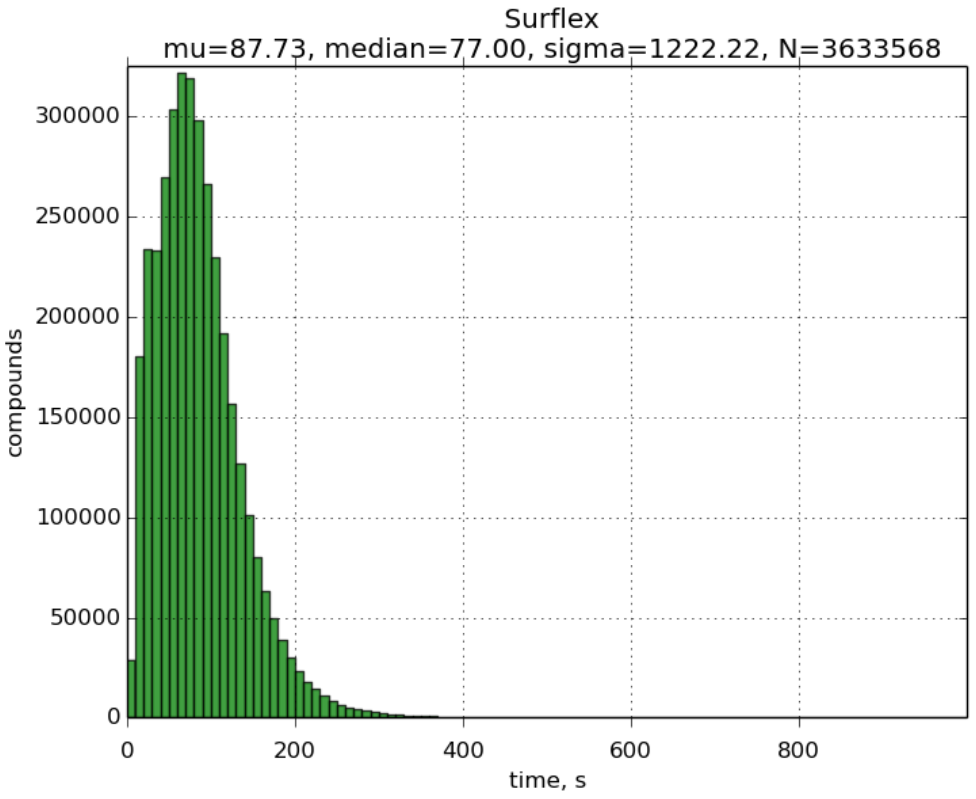


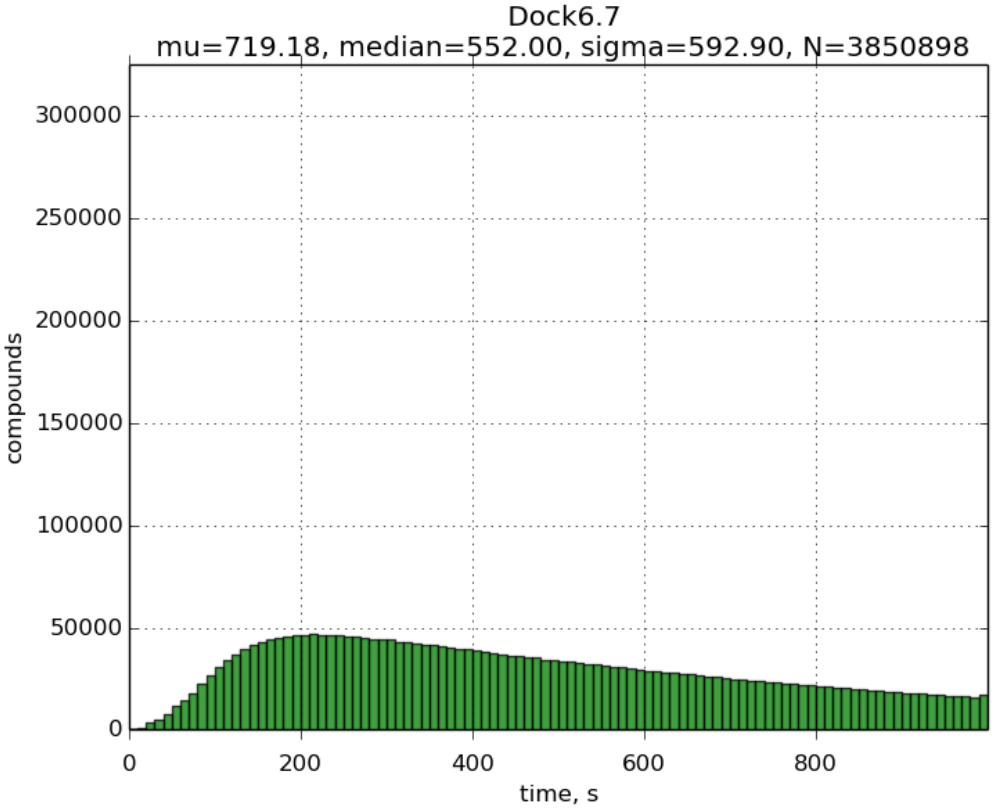



Estrogen_Receptor_1_Benchmarking
inactives
N = 20663
μ = −8.14
median = −8.10
σ = 1.03
actives
N = 383
μ = −9.93
median = −9.80
σ = 1.48
Counts
Counts
Vina Affinity (kcal/mol)
900
800
700
600
500
400
300
200
100
0
30
25
20
15
10
5
0
−14
−12
−10
−8
−6
−4

How expensive is docking?

- Compute time for docking depends the search space, search quality, and complexity of the scoring function.
- To dock millions of compounds, we cut corners.
- Docking time varies between programs (~1 minute/compound).





How do we scale to HTC resources?

- Each docking run is independent--*pleasantly parallelizable*!
- Typical docking codes do not benefit from specialized hardware or running on multiple cores.
- To maximize throughput:
  - Enable “Flock” and “Glide” to get access to more nodes.
  - Split compound library up into small chunks. Number of compounds is selected such that it should run in ~2hr for a given docking program. Due to variation in speed for different programs, this chunk size varies from 5 and 500 compounds!
  - Each chunk is docked using a single slot—to scavenge ANY open slots. Dock compounds within chunk serially.
  - Checkpointing is enabled and a wrapper script is used to track the compounds completed in case job is evicted and migrates to another node.

How do we benefit from HTC?

- Very large number of compounds
- Large numbers of targets
- Extensive docking parameter testing
- Benchmarking of different programs
- Hypothetical 100 node cluster = 3.5 million/day
- Local SMSF (3 nodes) = 35,000/day
- 76 million dockings!
- 4.6 million weighted CPU hours used

What have we learned with HTC resources?

Fold-enrichment for actives in top 100 ranking molecules:

	#mols	#actives	AD4	Dock6.7	Fred	Hybrid	Plants	rDock	Smina	Surflex	Consensus	Hits
adam17	36316	532	0.0	0.0	19.8	28.7	5.5	15.0	17.7	8.2	36.2	53
esr2	20513	367	27.7	14.9	52.0	52.5	17.3	28.4	29.1	15.7	44.7	80
glcm	3850	54	2.8	5.9	4.3	20.0	8.6	11.8	2.1	17.1	19.3	27
hivint	6736	100	0.9	8.3	6.7	6.7	12.8	5.1	6.7	4.0	14.1	21
hivpr	36156	535	0.7	13.4	6.8	12.2	22.3	5.7	5.4	19.6	27.0	40
plk1	6896	107	5.8	1.3	8.4	8.4	1.3	7.0	0.0	0.0	3.9	6
try1	26320	449	9.4	19.3	27.6	29.9	23.4	19.6	1.8	48.7	39.9	68
Average:	**19541**	**306**	6.7	9.0	17.9	22.6	13.0	13.3	9.0	16.2	26.4	41

Thank You!

- UWCCC-Drug Discovery Core
- Mike Hoffmann
- Scott Wildman & Ken Satyshur
- Michael Newton & Tony Gitter
- OpenScienceGrid & CHTC
- Facilitators: Lauren Michael & Christina Koch

Carbone Cancer Center
UNIVERSITY OF WISCONSIN
SCHOOL OF MEDICINE AND PUBLIC HEALTH

CENTER FOR
HIGH THROUGHPUT
COMPUTING

Open Science Grid

Appendix

```
.
├── esr2-0001
│   ├── CHEMBL101382_01_esr2.mol2
│   ├── CHEMBL101914_01_esr2.mol2
│   ├── CHEMBL1083178_01_esr2.mol2
│   ├── CHEMBL1083981_01_esr2.mol2
│   ├── CHEMBL1086644_01_esr2.mol2
│   ├── CHEMBL1087683_01_esr2.mol2
│   ├── CHEMBL1087812_01_esr2.mol2
│   ├── CHEMBL1098710_01_esr2.mol2
│   └── lig.list
├── esr2-0002
│   ├── CHEMBL152970_01_esr2.mol2
│   ├── CHEMBL153113_01_esr2.mol2
│   ├── CHEMBL153303_01_esr2.mol2
│   ├── CHEMBL153395_01_esr2.mol2
│   ├── CHEMBL153405_01_esr2.mol2
│   ├── CHEMBL153471_01_esr2.mol2
│   ├── CHEMBL153706_01_esr2.mol2
│   ├── CHEMBL153765_01_esr2.mol2
│   └── lig.list
├── esr2-0003
│   ├── CHEMBL184367_00_esr2.mol2
│   ├── CHEMBL184371_00_esr2.mol2
│   ├── CHEMBL184421_00_esr2.mol2
│   ├── CHEMBL184598_00_esr2.mol2
│   ├── CHEMBL184598_01_esr2.mol2
│   ├── CHEMBL184598_02_esr2.mol2
│   ├── CHEMBL184958_00_esr2.mol2
│   ├── CHEMBL184958_01_esr2.mol2
│   └── lig.list
└── shared
    ├── count.log
    ├── p1-esr2-protomol.mol2
    ├── recp_esr2_H.mol2
    ├── surflex_bin.lic
    ├── surflex-dock-v3040-linux64.exe
    └── surflex_run_DUDE_v1.8_esr2.sh
```

```
#!/bin/bash

export SURFLEXLIC=`pwd`"/surflex_bin.lic"

targname=esr2

# bash surflex wrapper to keep a log of completed runs within a folder of ligands
# this will allow us to checkpoint on HTCondor and move our output when we
# get evicted from an execute node.

# set the current count based on last line of the count.log file
count=`tail -1 count.log`
targ=recp_${targname}_H.mol2
totalmols=`wc -l lig.list | awk '{print $1}'`
# loop through 50 small molecules
while [ $count -le $totalmols ]; do
    # lig.list is provided in each lig-### folder
    lig=`cat lig.list | sed -n "${count}p"`
    ligname=${lig%.mol2}
    time1=$(date +%s)
    ./surflex-dock-v3040-linux64.exe -pgeom \
                                     -multistart 4 \
                                     -ndock_final 1 \
                                     -spindense 6.00 \
                                     dock \
                                     $lig \
                                     p1-${targname}-protomol.mol2 \
                                     ${targ} \
                                     > log_${ligname}_${targname}.txt 2>&1
    time2=$(date +%s)
    elapsed=$(($time2-$time1))
    echo "surflex-dock Run:$count Time elapsed:$elapsed" >> time_elapsed.log
    mv sfdock-log sfdock-log-${ligname}_${targname}
    mv sfdock-log-results.mol2 sfdock-log-results-${ligname}_${targname}.mol2
    mv sfdock-log-results_tab.log sfdock-log-results_tab-${ligname}_${targname}.log
    let count=count+1
    echo $count >> count.log
done

# get important output
grep crash log_*_${targname}.txt > surf_scores.txt
cat sfdock-log-results-*.mol2 > poses.mol2

# clean up
rm sfdock-*
rm log_*_${targname}.txt
```