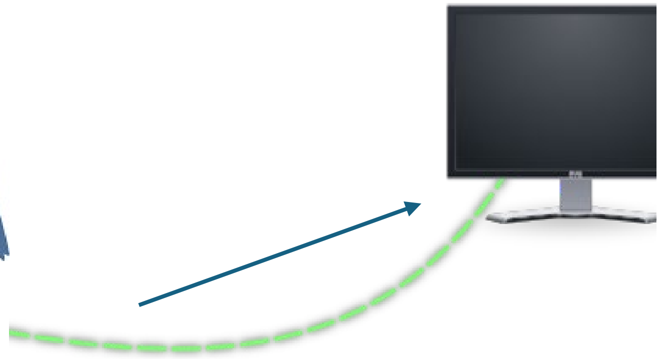
Using MD engines to stream simulation output to the Client

LAMMPS

FAST. FLEXIBLE. FREE. GROMACS

NAMD Scalable Molecular Dynamics



Amruthesh Thirumalaiswamy

ASU School of Molecular Sciences
Arizona State University

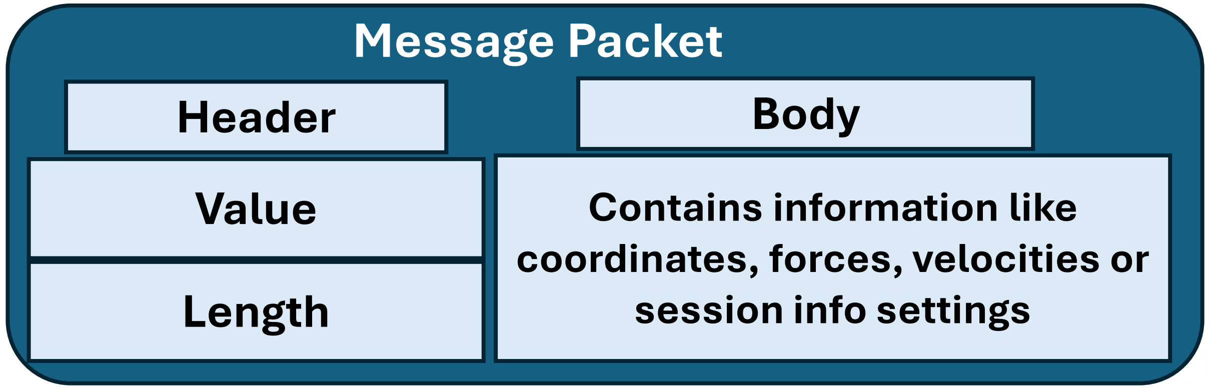
IMDv3 functionality in various MD engines

- **Source Code modifications**
  - IMDv3 features were appended to existing code modules pertaining to IMDv2
  - The IMDv2 code template was maintained and only changed where needed
  - The modified source codes have been tested for consistency and against unchanged versions of the codebases.

- **IMDv3 functionality**
  - IMDv3 features were added in accordance with the new protocol
  - Functions and modules added -- provide the ability to stream (send) specific information
  - Ability to control streaming settings through MD engine input configuration file

Streaming Data

- Data is streamed and sent in the from of packets which contain a ***header*** and a ***body***


Message Packet
Header
Body
Value
Length
Contains information like coordinates, forces, velocities or session info settings

- **Value** sets the type of Message/information being sent in the body
- **Length** sets the length of that information
- Simple messages like **IMD_GO, IMD_DISCONNECT, IMD_KILL** do not have a body and have a zero length
- Others like **IMD_FCOORDS** or **IMD_VELCOITIES** have long array-like data within their bodies
- **IMD_HANDSHAKE** -- special case i.e. length variable contains the *IMD version* number

IMD Message Type	enum value
IMD_DISCONNECT	0
IMD_ENERGIES	1
IMD_FCOORDS	2
IMD_GO	3
IMD_HANDSHAKE	4
IMD_KILL	5
IMD_MDCOMM	6
IMD_PAUSE	7
IMD_TRATE	8
IMD_IOERROR	9
IMD_SESSIONINFO	10
IMD_RESUME	11
IMD_TIME	12
IMD_BOX	13
IMD_VELOCITIES	14
IMD_FORCES	15
IMD_WAIT	16

Configuring MD engine input for streaming

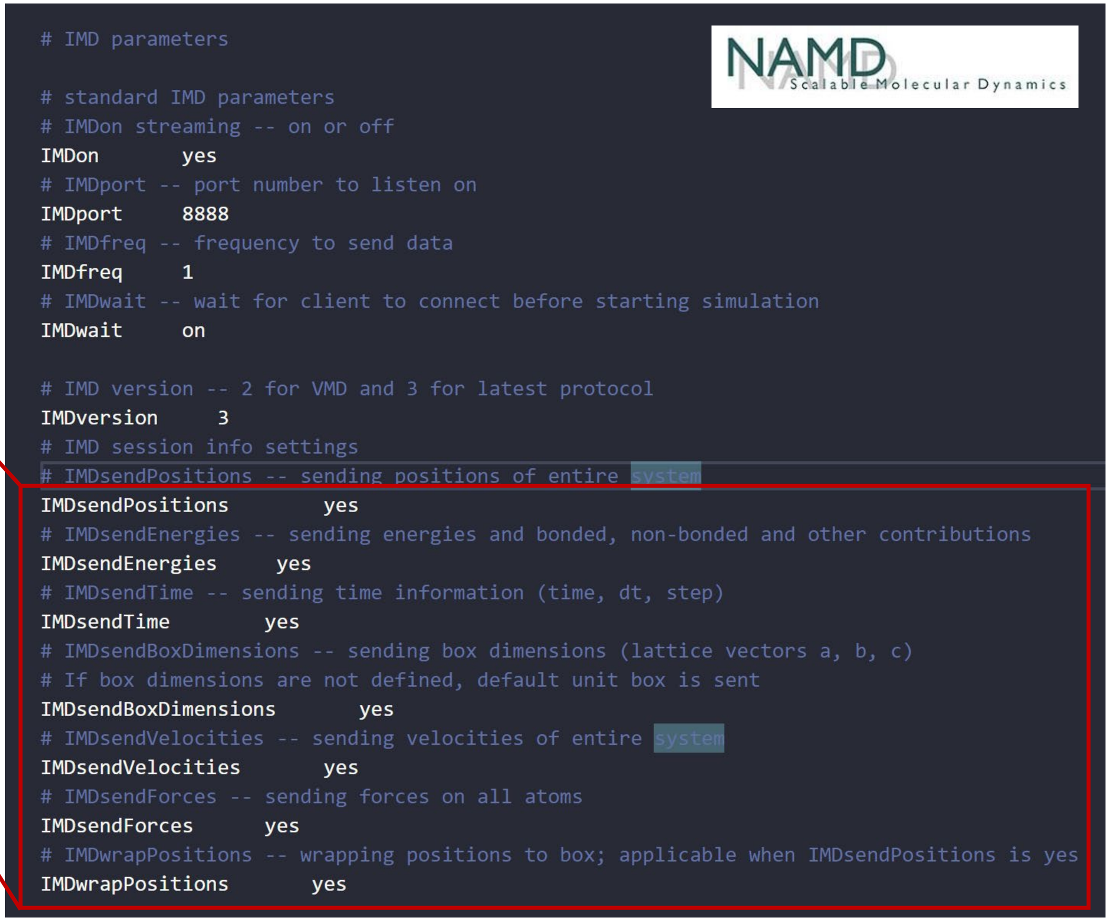
- **IMD_SESSIONINFO** -- special message packet, contains setting options for streaming process
- Settings configured by the user in simulation input file

```
Header:
    10 (int32) Value: Session info
    7  (int32) Length: Number of configuration options in the body

Body:
    <val> (int8) Are time packets sent?
    <val> (int8) Are IMD energy block packets sent?
    <val> (int8) Are box packets sent?
    <val> (int8) Are coordinate packets sent?
    <val> (int8) Are coordinates wrapped into the simulation box?
                 Meaningless if coordinates not sent!
    <val> (int8) Are velocity packets sent?
    <val> (int8) Are force packets sent?
```

- Similar input functionality available in other MD engines code packages

- As an example, NAMD uses an input configuration file that would look like ..



```
# IMD parameters

# standard IMD parameters
# IMDon streaming -- on or off
IMDon        yes
# IMDport -- port number to listen on
IMDport      8888
# IMDfreq -- frequency to send data
IMDfreq      1
# IMDwait -- wait for client to connect before starting simulation
IMDwait      on

# IMD version -- 2 for VMD and 3 for latest protocol
IMDversion     3
# IMD session info settings
# IMDsendPositions -- sending positions of entire system
IMDsendPositions        yes
# IMDsendEnergies -- sending energies and bonded, non-bonded and other contributions
IMDsendEnergies     yes
# IMDsendTime -- sending time information (time, dt, step)
IMDsendTime        yes
# IMDsendBoxDimensions -- sending box dimensions (lattice vectors a, b, c)
# If box dimensions are not defined, default unit box is sent
IMDsendBoxDimensions       yes
# IMDsendVelocities -- sending velocities of entire system
IMDsendVelocities       yes
# IMDsendForces -- sending forces on all atoms
IMDsendForces      yes
# IMDwrapPositions -- wrapping positions to box; applicable when IMDsendPositions is yes
IMDwrapPositions       yes
```

Configuring MD engine input for streaming

LAMMPS

- The input file in LAMMPS can be set with a line that takes care of various IMDv3 settings

```
## IMD settings
# https://docs.lammps.org/fix_imd.html
fix 2 all imd 8888 trate 1 version 3 unwrap off time on box on coordinates on velocities on forces on
```

FAST. FLEXIBLE. FREE.
GROMACS

- For GROMACS the mdp file is modified with requisite IMDv3 settings, which can then be assembled into a binary tpr using grompp

```
; IMD parameters
IMD-group                = System      ; group to send to IMD
IMD-nst                  = 1           ; transmssion rate to IMD
IMD-version              = 3           ; version of IMD protocol
IMD-time                 = yes         ; if time information should be sent (time, dt, step)
IMD-box                  = yes         ; if box dimensions should be sent (lattice vectors a, b, c)
IMD-coords               = yes         ; if coordinates should be sent
IMD-unwrap               = no          ; if coordinates should be unwrapped
IMD-vels                 = yes         ; if velocities should be sent
IMD-forces               = yes         ; if forces should be sent
IMD-energies             = no          ; if energies should be sent
```

Availability of Source Codes and Docker containers

LAMMPS

- GitHub - ljwoods2/lammps: Public development project of the LAMMPS MD software package
- https://github.com/ljwoods2/lammps

NAMD

- GitHub - amruthesht/namd-3.0-IMDv3-patch
- https://github.com/amruthesht/namd-3.0-IMDv3-patch

Patch with instructions

GROMACS

- Files · imd-v3 · ljwoods2 / GROMACS · GitLab
- https://gitlab.com/ljwoods2/gromacs/-/tree/imd-v3?ref_type=heads

Docker Containers

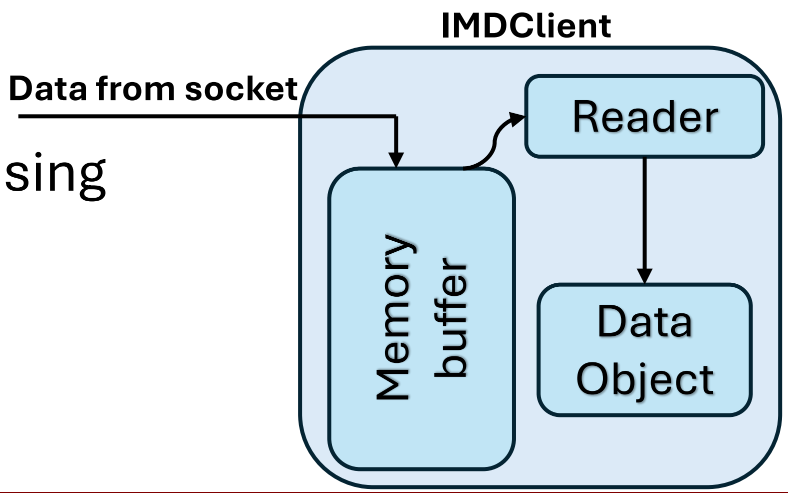
- GitHub - Becksteinlab/streaming-md-docker: Docker images with MD packages that support streaming with IMD v3.
- https://github.com/Becksteinlab/streaming-md-docker

IMDClient – the receiver

- The receiver is any program capable of receiving data from the IMD stream
- IMDv2 used a VMD as a receiver VMD Visual Molecular Dynamics

GitHub - Becksteinlab/imdclient: Streaming analysis from running MD simulations.

- **IMDClient** -- python package that is compatible with the IMDv3 protocol
  - built to receive and process incoming streaming data from producer (MD engine)
- Provides *infrastructure* to read data from the socket
  - Data is read from socket – stored in a temporary buffer
- Data from buffer is read into an object for further processing and analysis



IMDClient – the receiver

- Contains a **IMDReader** class built upon MDAnalysis reader base class
  - Data from buffer read into MDAnalysis universe
- Data can be paired with different MDAnalysis functions to perform analysis

IMDClient

- GitHub - Becksteinlab/imdclient: Streaming analysis from running MD simulations.
- IMDclient is publicly available here: https://github.com/Becksteinlab/imdclient
- It is available via PyPI and can be easily installed with pip as follows:

```
pip install imdclient
```

- And can be done inside a virtual environment in conda

```
conda create --name imdclient
conda activate imdclient
pip install imdclient
```

This framework allows us to perform ***on-the-fly analysis***

On the fly analysis

- **Producer**: MD simulation run on **NAMD** for a large system describing a lipid membrane with ions permeating through it (run on remote server)

- **Receiver**: IMDClient run on Jupyter Notebook with MDAnalysis to receive and perform in-situ **RMSF** and **RDF** analysis and provide a **live visualization** (run on remote server – separate core)