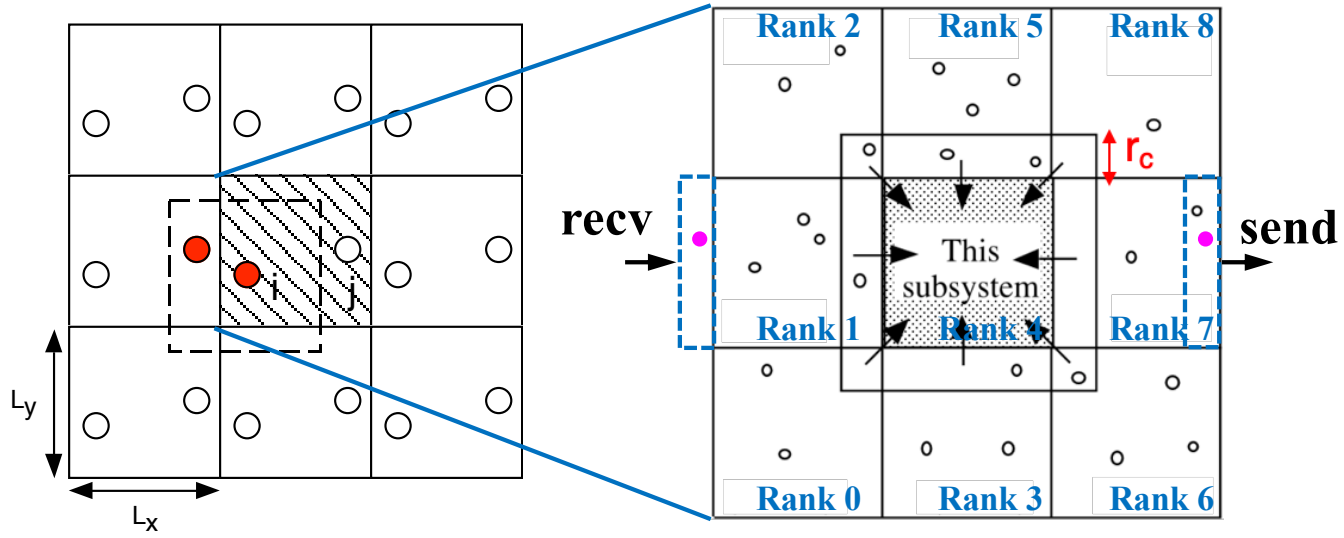
Parallel Molecular Dynamics: FAQ

Q. Where is periodic boundary condition in pmd.c?

A. It's implicitly implemented in atom_copy() & atom_move()

1. Interaction computation (minimum-image convention) in atom_copy()



https://aiichironakano.github.io/cs596/01MD.pdf

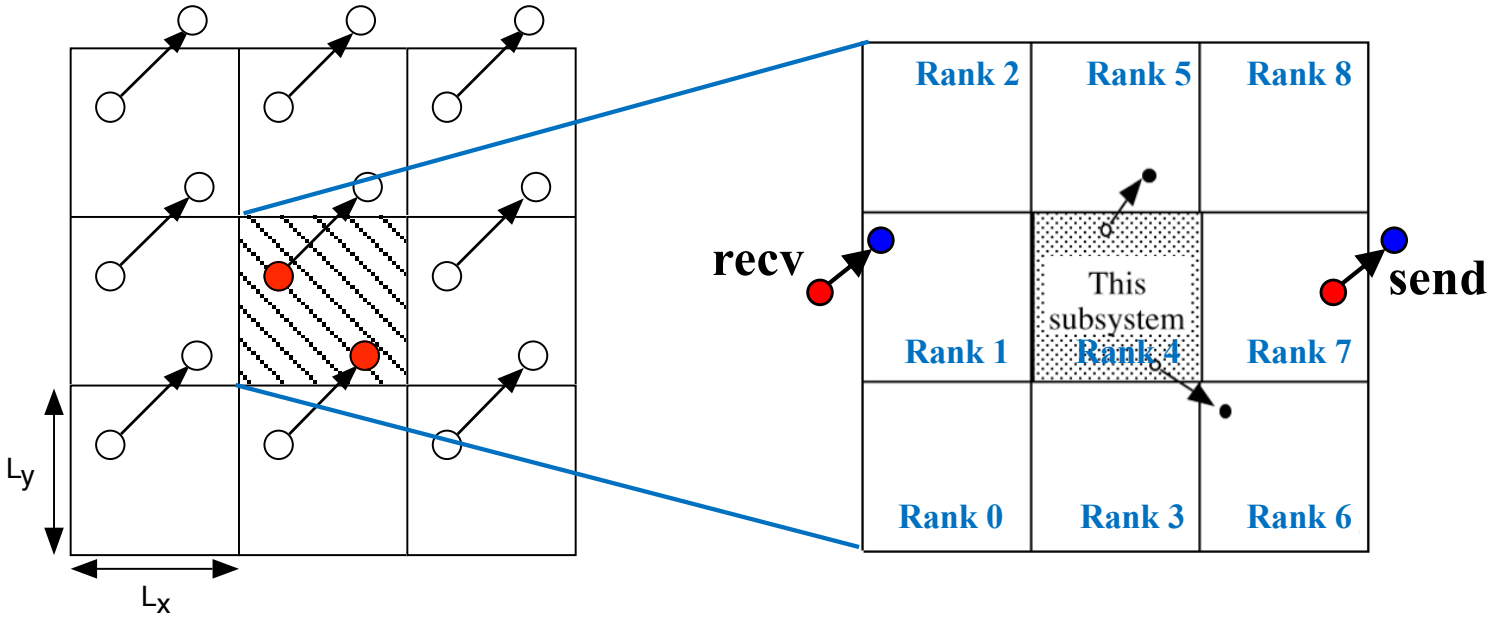
https://aiichironakano.github.io/cs596/03ParallelMD.pdf

Parallel Molecular Dynamics: FAQ

Q. Where is periodic boundary condition in pmd.c?

A. It's implicitly implemented in atom_copy() & atom_move()

2. Fold back to central image in atom_move()



https://aiichironakano.github.io/cs596/01MD.pdf https://aiichironakano.github.io/cs596/03ParallelMD.pdf

Parallel Molecular Dynamics: FAQ

Q. **Why define vector process ID?**

A. **Vector process ID (`vid[3]`) is used only to find the ranks (`nn[6]`) of the six neighbor processes during the initialization of the run, in function `set_topology()`**

```
/* Integer vectors to specify the six neighbor ranks */
 int iv[6][3] = { {-1,0,0},{1,0,0},{0,-1,0},{0,1,0},{0,0,-1},{0,0,1} };
 int ku,a,k1[3];
 /* Set up neighbor tables, nn & sv */
 for (ku=0; ku<6; ku++) {
   /* Vector index of neighbor ku (with wrap-around condition) */
   for (a=0; a<3; a++)
     k1[a] = (vid[a]+iv[ku][a]+vproc[a])%vproc[a];
   /* Scalar neighbor ID, nn */
   nn[ku] = k1[0]*vproc[1]*vproc[2]+k1[1]*vproc[2]+k1[2];
   /* Shift vector, sv */
   for (a=0; a<3; a++) sv[ku][a] = al[a]*iv[ku][a];
 }
 /* Set up the node parity table, myparity */
 for (a=0; a<3; a++) {
   if (vproc[a] == 1)
     myparity[a] = 2;
   else if (vid[a]%2 == 0)
     myparity[a] = 0;
   else
     myparity[a] = 1;
 }
```

Serial process ID

In `main()`:

```
MPI_Comm_rank(MPI_COMM_WORLD, &sid);
vid[0] = sid/(vproc[1]*vproc[2]);
vid[1] = (sid/vproc[2])%vproc[1];
vid[2] = sid%vproc[2];
```

Parallel Molecular Dynamics: FAQ

Q. **Why accelerate the velocity for half time-step in the velocity Verlet algorithm, instead of full time-step as in Euler algorithm?**

A. **By doing so, a conservation law called Liouville's theorem (or phase-space volume conservation) is satisfied exactly by velocity Verlet (but not Euler) algorithm; this in turn leads to superior long-term stability of the trajectory in the former, reflected, *e.g.*, in better energy conservation**

See slides 25-28 in https://aiichironakano.github.io/phys516/02MD-slide.pdf

Mapping: $\overbrace{(x,p)}^{t} \rightarrow \overbrace{(x',p')}^{t+\Delta}$

Liouville's theorem: $\frac{\partial(x',p')}{\partial(x,p)} = 1$

In Euler algorithm:

$$\vec{v}_i(t+\Delta) \leftarrow \vec{v}_i(t) + \vec{a}_i(t)\Delta$$