PETSc DMNetwork: A Library for Scalable Network PDE-Based Multiphysics Simulation

Hong Zhang

Computer Science Dept., Illinois Institute of Technology
and
Mathematics and Computer Science Div.,
Argonne National Laboratory, USA

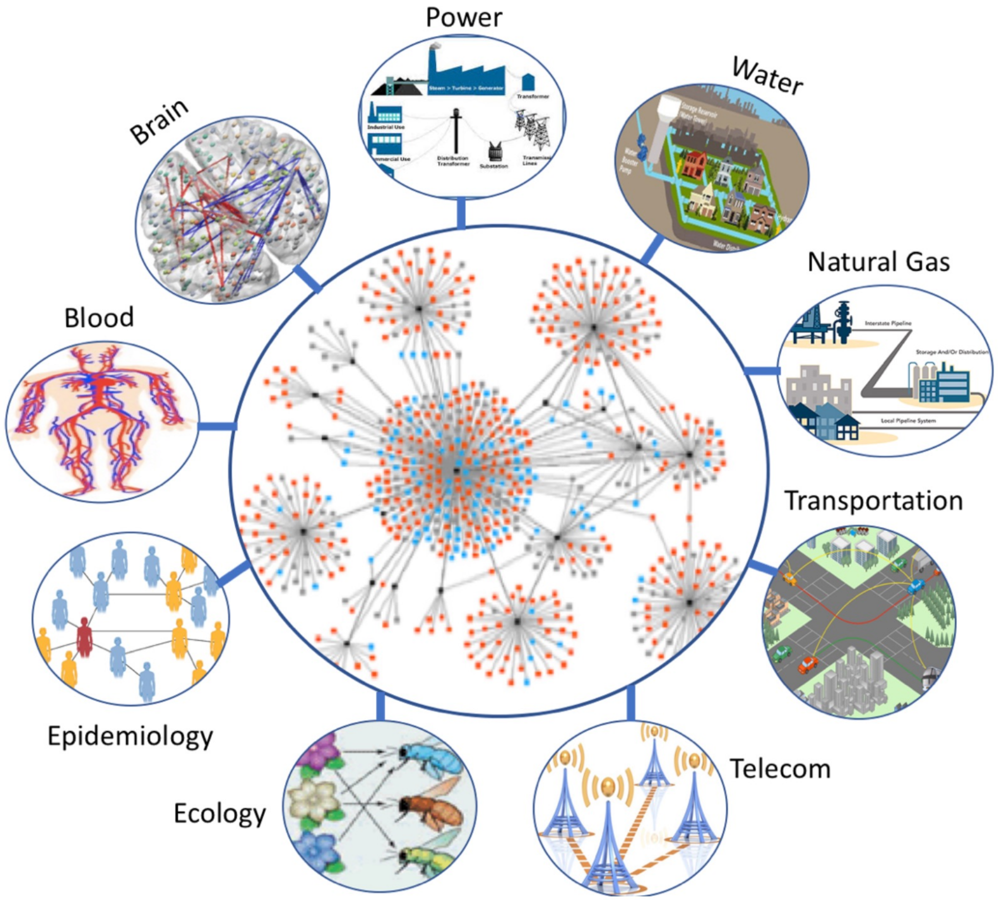
June 2023

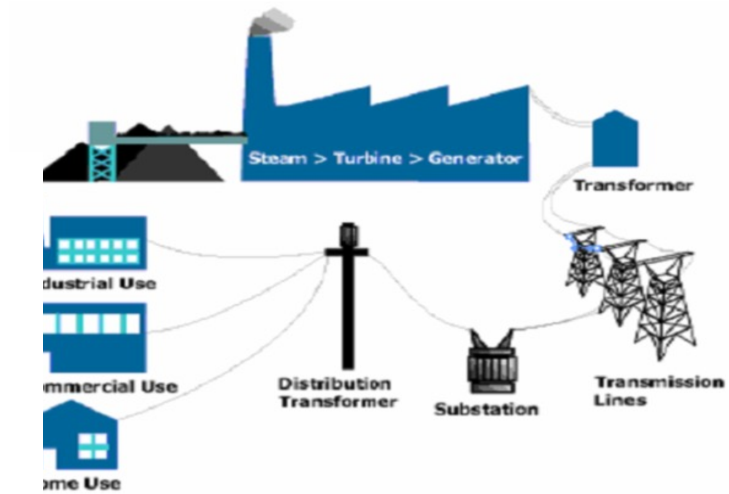
DMNetwork

https://www.mcs.anl.gov/petsc/dmnetwork/index.html

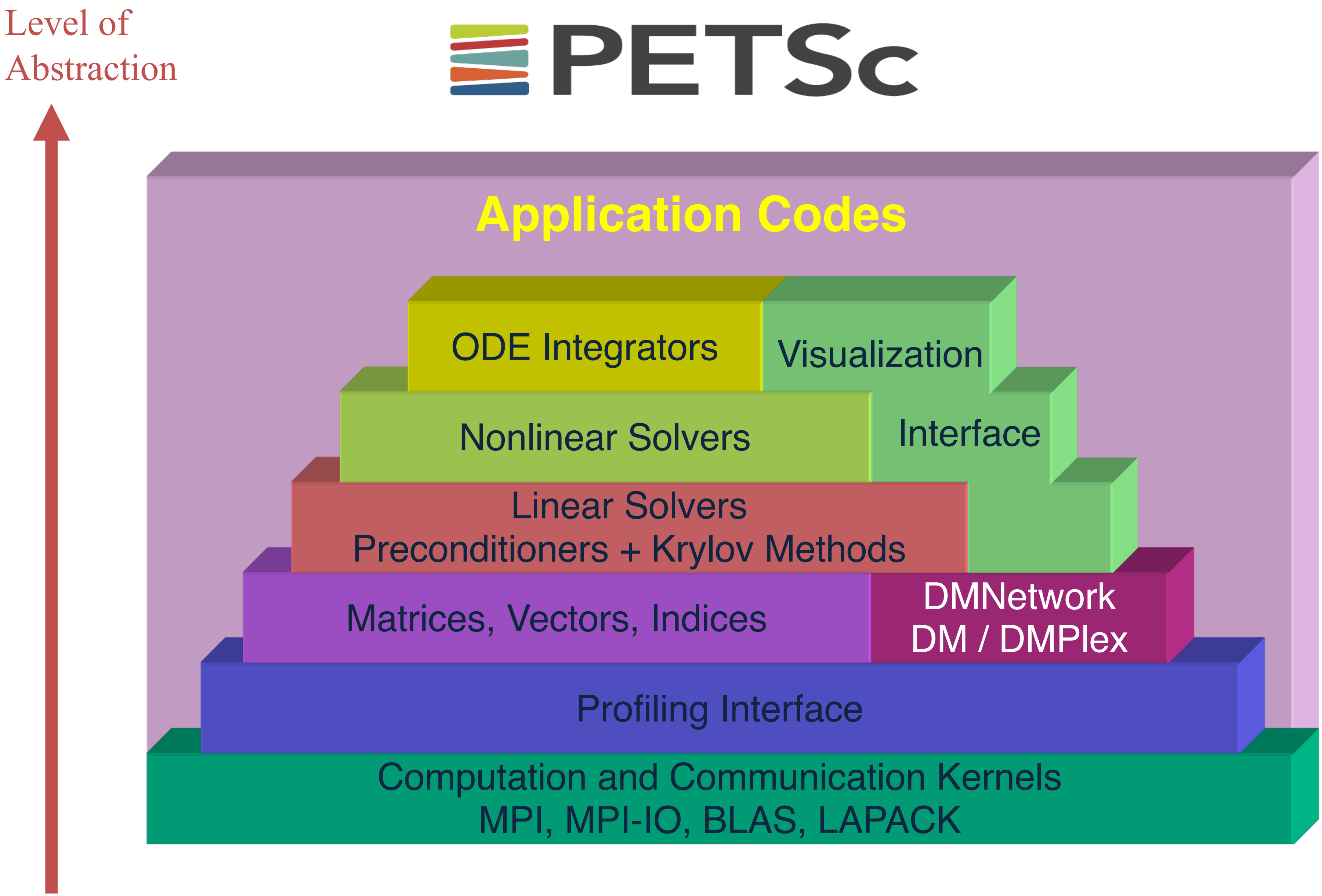
- a network multiphysics simulation package in PETSc
- data and topology management
- scalable, hierarchical and composable solvers
- eases the simulation development cycle by providing the infrastructure through simple abstractions to define and query the network components.
- PETSc DMNetwork: A Library for Scalable Network PDE-Based Multiphysics Simulations. Abhyankar S., Betrie G., Maldonado D.A, McInnes L.C., Smith B., Zhang H. , ACM Transactions on Mathematical Software, Vol. 46, No.1, Article 5, 2020.

Multiphysics Network Applications



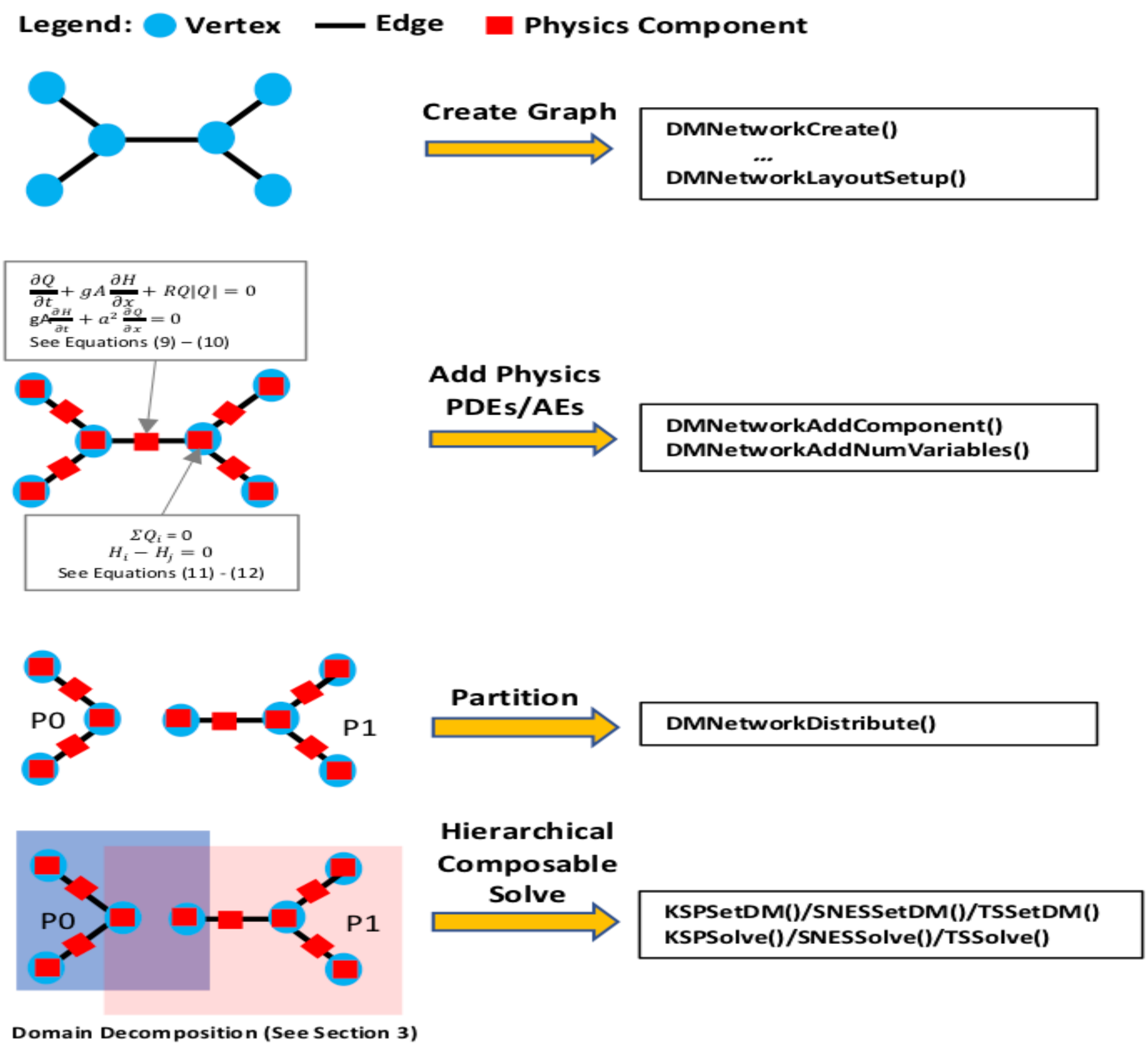


PETSc

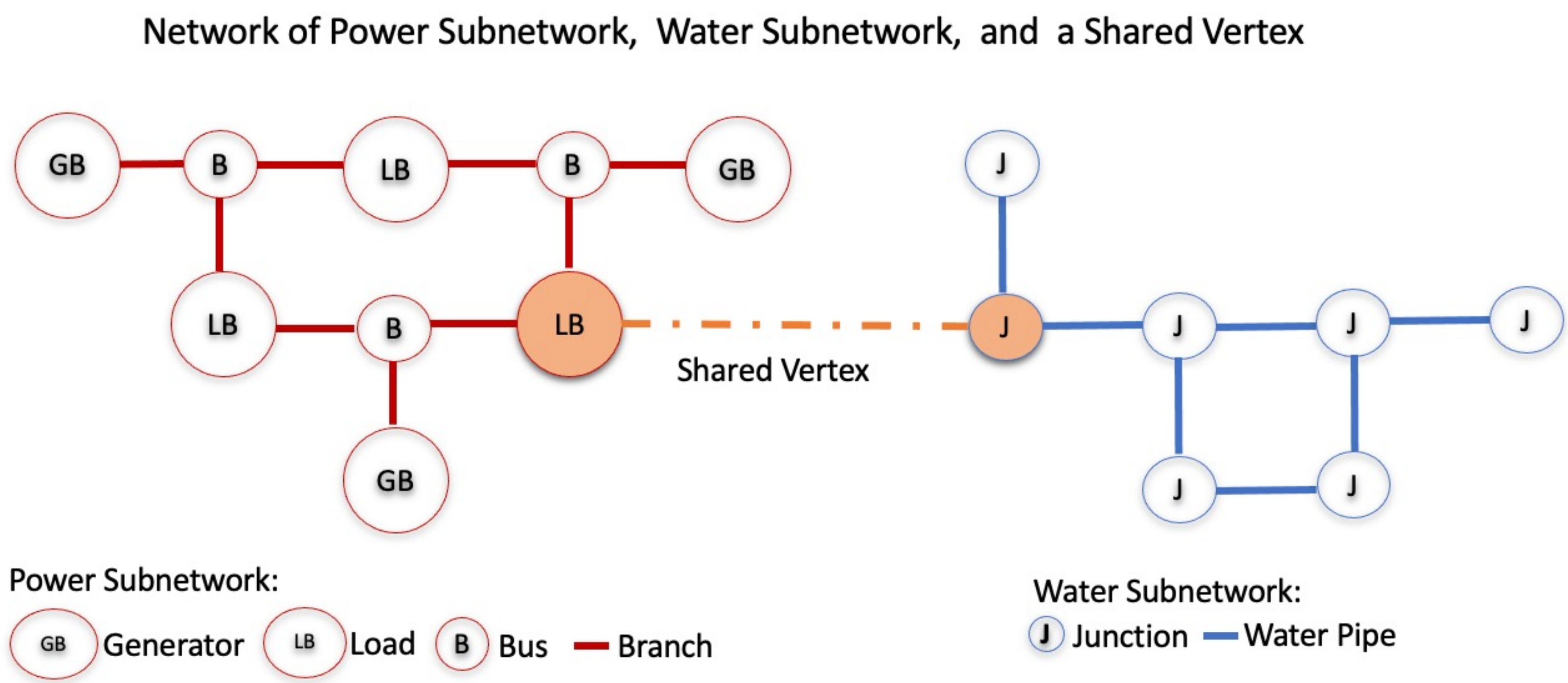


PETSc Structure

DMNetwork Application Steps



Example: petsc/src/snes/tutorials/network/ex1.c



Example: petsc/src/snes/tutorials/network/ex1.c

Let

$$X = \begin{bmatrix} X_{power} \\ X_{water} \end{bmatrix}, \quad F(X) = \begin{bmatrix} F_{power}(X_{power}) \\ F_{water}(X_{water}) \\ F_{couple}(X) \end{bmatrix}. \tag{5}$$

The nonlinear mathematical system we wish to solve is

$$F(X) = 0. \tag{6}$$

We create three nonlinear solver objects: SNES, SNES_power, and SNES_water. SNES solves the coupled Equation (6). SNES_power solves

$$\begin{bmatrix} F_{power}(X_{power}) \\ X_{water} - X_{water_{old}} \end{bmatrix} = 0, \tag{7}$$

and SNES_water solves

$$\begin{bmatrix} F_{water}(X_{water}) \\ X_{power} - X_{power_{old}} \end{bmatrix} = 0, \tag{8}$$

Example: petsc/src/snes/tutorials/network/ex1.c

```
/* Create graph */
DMNetworkCreate(comm, &dm);

DMNetworkRegisterComponent(dm, "branch", sizeof(struct_Edge_Power), &power->branch);
DMNetworkRegisterComponent(dm, "bus", sizeof(struct_VERTEX_Power), &power->bus);
DMNetworkRegisterComponent(dm, "vertex_water", sizeof(struct_VERTEX_Water), &water->vtx);

DMNetworkSetNumSubNetworks(dm, PETSC_DECIDE, Nsubnet);

DMNetworkAddSubnetwork(dm, "power", nVertices[0], nEdges[0], edgelist_power, &power_netnum);
DMNetworkAddSubnetwork(dm, "water", nVertices[1], nEdges[1], edgelist_water, &water_netnum);

DMNetworkAddSharedVertices(dm, power_netnum, water_netnum, 1, &power_svtx, &water_svtx);

DMNetworkLayoutSetUp(dm);
```

Example: petsc/src/snes/tutorials/network/ex1.c

```
/* Add components and num of variables to the power subnetwork */
DMNetworkGetSubnetwork(dm, power_netnum, &nv, &ne, &vtx, &edges);
for (i = 0; i < nv; i++) {
   DMNetworkIsSharedVertex(dm, vtx[i], &flg);
   if (flg) continue;
   DMNetworkAddComponent(dm, vtx[i], power->compkey_bus, &bus[i], nvar_power);
}
/* Add components and num of variables to the water subnetwork */
…
/* Add components and num of variables to the shred vertex */
DMNetworkGetSharedVertices(dm, &nv, &vtx);
for (i = 0; i < nv; i++) {
   DMNetworkAddComponent(dm, vtx[i], power->compkey_bus, &bus[4],  nvar_power);
   DMNetworkAddComponent(dm, vtx[i], power->compkey_load, &load[0], 0)
   DMNetworkAddComponent(dm, vtx[i], water-> compkey_vtx, &vertex[0], nvar_water);
}

DMSetUp(dm);
```

Example: petsc/src/snes/tutorials/network/ex1.c

```
/* Partition  */
DMNetworkDistribute(&dm, 0);

/* Create and setup solvers snes, snes_power and snes_water */
SNESCreate(comm, &snes);
SNESSetDM(snes, dm);
SNESSetFunction(snes, F, FormFunction, &user);
…
SNESCreate(comm, &snes_power);
SNESSetDM(snes_power, dm);
SNESSetFunction(snes_power, F, FormFunction, &user);
…
SNESCreate(comm, &snes_water);
SNESSetDM(snes_water, dm);
SNESSetFunction(snes_water, F, FormFunction, &user);
…
```

```
/* Solve */
user.it = 0; reason = SNES_DIVERGED_DTOL;
while (user.it < it_max && reason<0) {
   user.subsnes_id = 0;          SNESSolve(snes_power, NULL, X);
   user.subsnes_id = 1;          SNESSolve(snes_water, NULL, X);
   user.subsnes_id = Nsubnet; SNESSolve(snes, NULL, X));
   user.it++;                     SNESGetConvergedReason(snes, &reason);
};
```

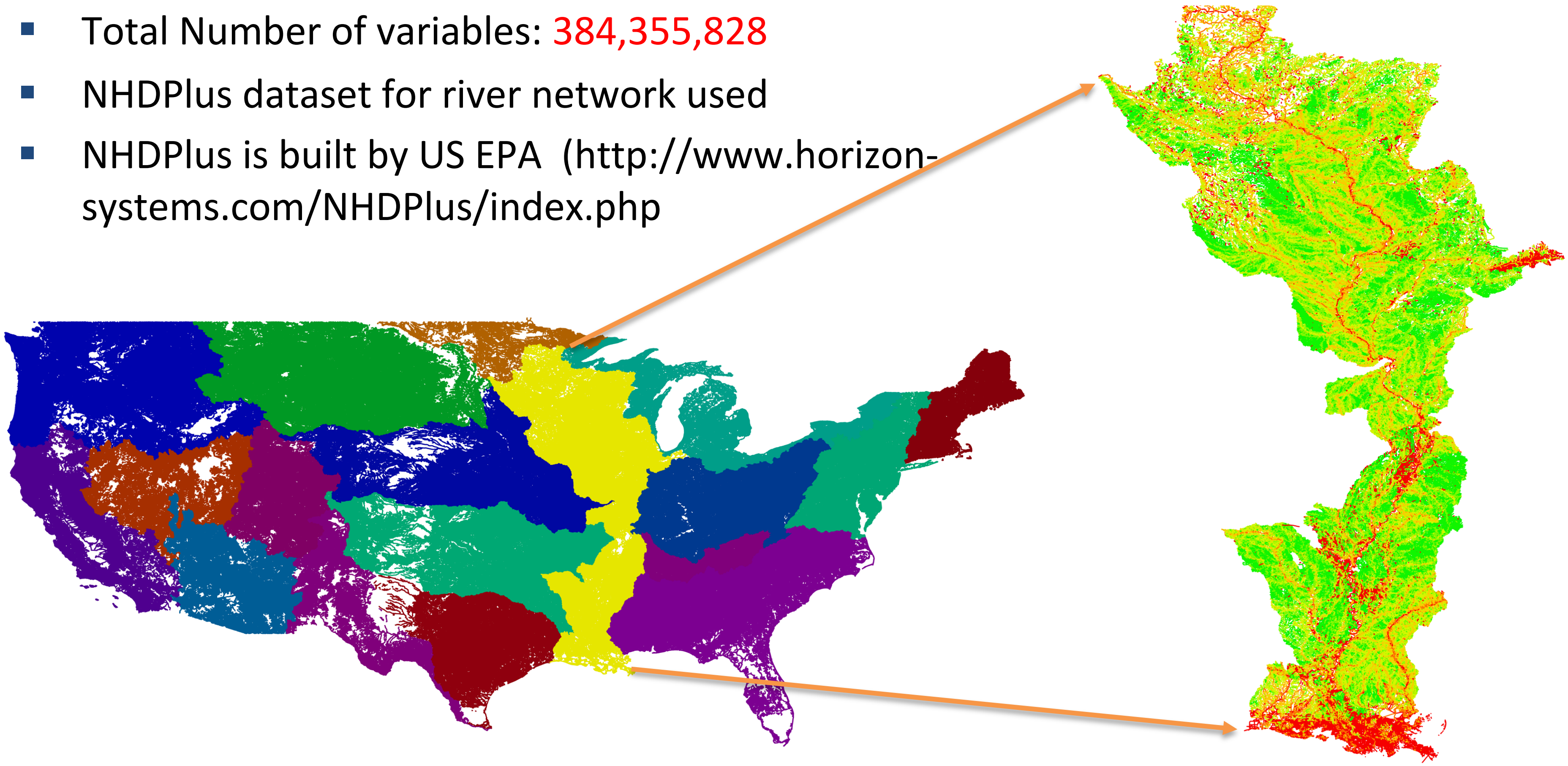
Running this code with runtime options, the users can experiment with domain decomposition and solver composition/splitting at various levels of physics and computation without modifying *ex1.c*. For example, for the overall coupled system (6), we use the following options.

```
-coupled_snes_fd
-coupled_ksp_type gmres
-coupled_pc_type bjacobi
-coupled_sub_pc_type lu
-coupled_sub_pc_factor_mat_ordering_type qmd

-power_pc_type asm
-power_sub_pc_type lu
-power_sub_pc_factor_mat_ordering_type qmd
```

```
/* Function evaluation */
FormFunction(SNES snes, Vec X, Vec F, void *appctx) {
…
 SNESGetDM(snes, &dm);
 DMGetLocalVector(dm, &localX); DMGetLocalVector(dm, &localF);
 DMGlobalToLocalBegin(dm, X,…, localX); DMGlobalToLocalEnd(dm, X,…, localX);

 /* Function evaluation on power subnetwork */
 DMNetworkGetSubnetwork(dm, 0, &nv, &ne, &vtx, &edges);
 if (user->subsnes_id == 1) { /* snes_water only */
  FormFunction_Dummy(dm, localX, localF, nv, ne, vtx, edges, user); /* localF(X) = localX - localXold */
 } else {
  FormFunction_Power(dm, localX, localF, nv, ne, vtx, edges, &appctx_power);
 }
 /* Function evaluation on water subnetwork */
 …
 /* Function evaluation at shared vertex */
 DMNetworkGetSharedVertices(dm, &nv, &vtx);
  …
}
```

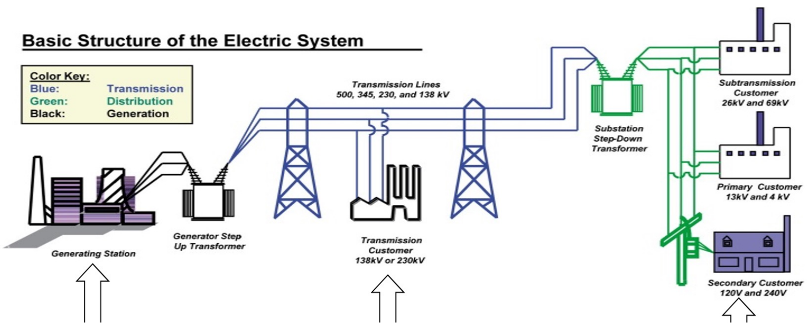
Continental US Major Rivers Simulation
(G. Betrie, H. Zhang)

- 3,098,638 reaches and 3,028,968 junctions
- Total Number of variables: 384,355,828
- NHDPlus dataset for river network used
- NHDPlus is built by US EPA (http://www.horizon-systems.com/NHDPlus/index.php



PETSc DMNetwork: A Library for Scalable Network PDE-Based Multiphysics Simulations, Abhyankar S., Betrie G., Maldonado D.A, McInnes L.C., Smith B., Zhang H. , ACM Transactions on Mathematical Software, 2020.

Parallel Primal-Dual Interior Point Method for Dynamic Optimal Power Flow



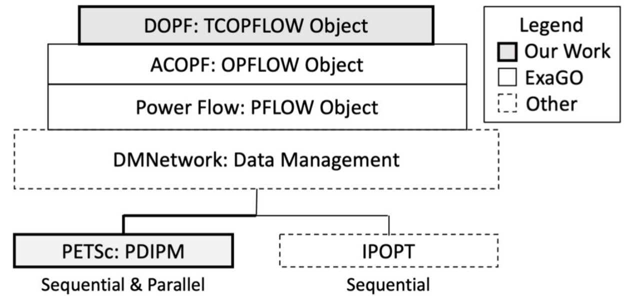
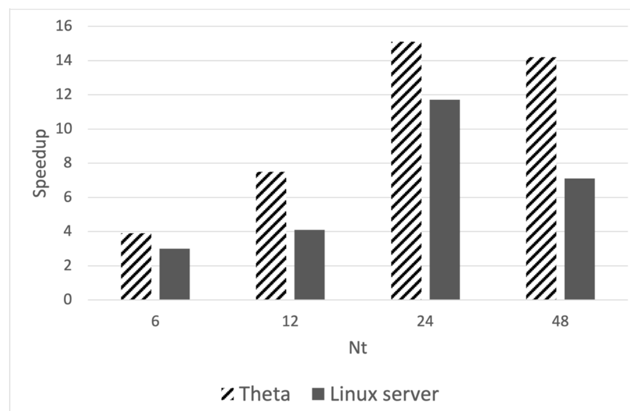


Fig. 2: Software structure of DOPF within ExaGO.

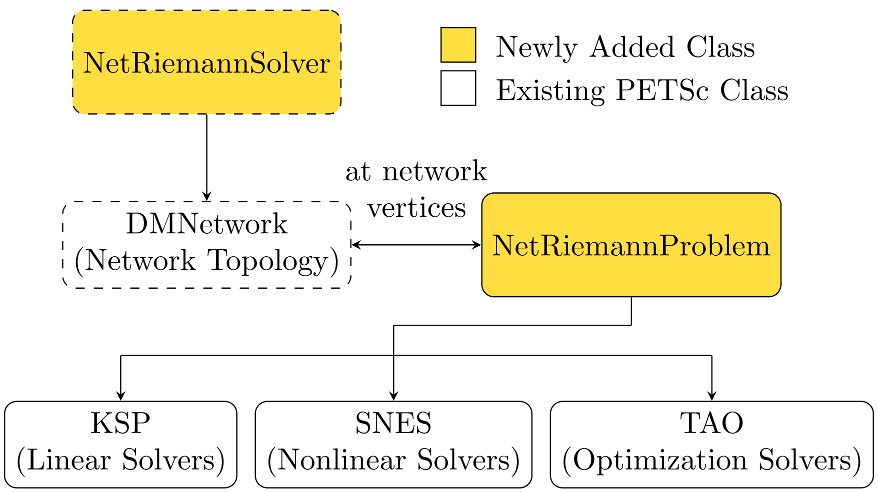
- Developed a new parallel primal-dual interior point solver in TAO
- Used multiphysics networks to exploit problem structure with a blocked preconditioner
- Applied to multiperiod optimal power flow networks with computational ease and efficiency
- Collaboration with PNNL for power-grid modeling, including members of the GridPACK and ECP ExaSGD projects

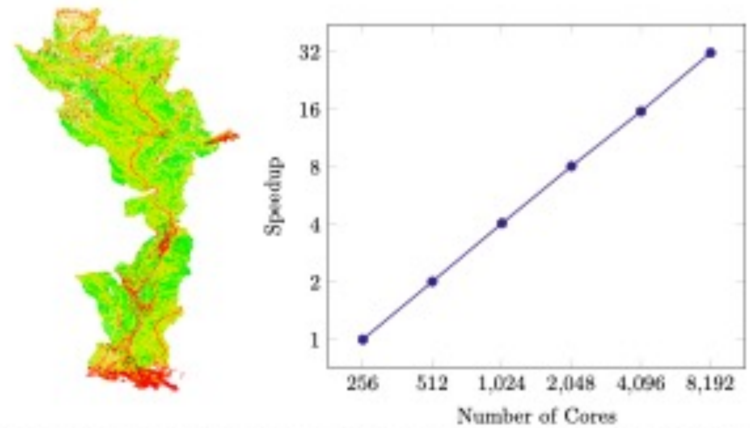


Parallel Primal-Dual Interior Point Method for the Solution of Dynamic Optimal Power Flow,
R. Sundermann, S. Abhyankar, H. Zhang, J. Kimn, T. Hansen, 2022, IET Generation, Transmission & Distribution.

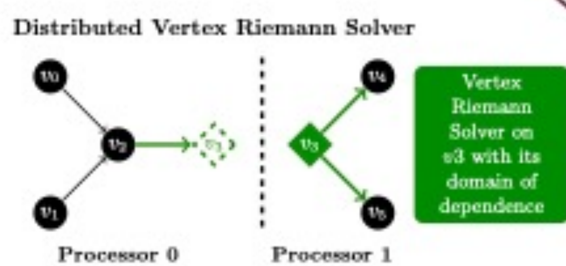
Scalable Riemann Solvers For Hyperbolic Network Simulations

Scalable vertex Riemann solvers for the simulation of hyperbolic conservation laws on networks, using discontinuous Galerkin discretization for network edges





Scalability of a Mississippi River network simulation.



Water height in network dam break

A. Hamilton, J. Qiu, H. Zhang, “Scalable Riemann Solvers with the Discontinuous Galerkin Method for Hyperbolic Network Simulations,” Proceedings of the PASC Conference, 2023.

Traffic Flow:
Data-Intensive Computation

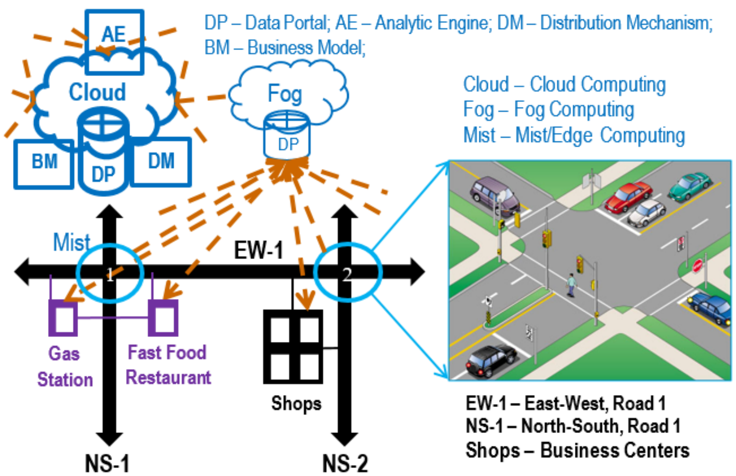


Figure 1: A representative GCI system

Total Profit: $P_{T\text{-}t} = P_{Sale\text{-}t} + P_{Staffing\text{-}t} + P_{Waste\text{-}t}$

Approaches:
1) OpenStreetMap and Sumo:
2) Google Maps Platform:
3) Location IQ

- Geospatial Cyberinfrastructure for Regional Economic Growth
- Exploring new capabilities motivated by the needs of data-driven computing.
- Collaboration with Prof. A. Asaduzzaman of WSU via SHI SRP-HPC PROGRAM
- Student participants: D. Campbell, N. Nawal.

New addition: PetscViewer with CSV format

-- requires **matplotlib** and **pandas**

(D. Campbell, A. Hamilton, H. Zhang)

Example: petsc/src/snes/tutorials/network/ex1.c

```
mpiexec -n 3 ./ex1
-petscpartitioner_type parmetis
-dmnetwork_view draw
-dmnetwork_view_distributed draw
-dmnetwork_view_all_ranks
```

```
./ex1 -monitorIteration -monitorColor
-power_snes_max_it 0
-water_snes_max_it 0
-coupled_snes_max_it 10
-draw_pause 5.0
```