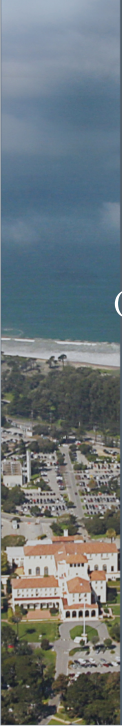


Optimization via Adaptive Resolvent Splitting

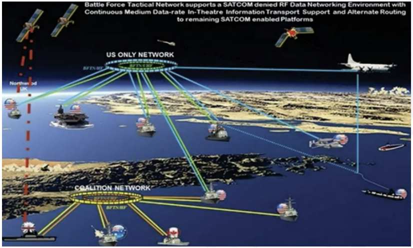
Peter Barkley

Naval Postgraduate School

January 18, 2024

Motivation

- Many optimization problems remain too big for one CPU
- Thankfully, we have lots of CPUs available
- Decomposing, or "splitting," the problem, breaks it into subproblems which are individually tractable
- This has to be done thoughtfully to actually converge
- Many existing algorithms fail to account for the communication structure between the compute nodes



Goal

Distributed, decentralized optimization of finite sums of convex functions via splitting methods which can be optimally adapted for the communication structure of the compute nodes and the structure of the problem.

Focusing on the following contexts:

1. Autonomous UxS mission optimization
   – "1,000 targets in 24 hours"
2. Very large scale optimization problems in a high performance parallel computing environment

Agenda

- Weapon Target Allocation Problem
- Mathematical Foundations
- Algorithm Design
- Future Work

Weapon Allocation

- The mission is to optimally assign various munitions to a set of different targets
- Munitions have different probability of kill (Pk) for each target
- Each target has an assigned value
- An optimal assignment minimizes the value of the surviving enemy units

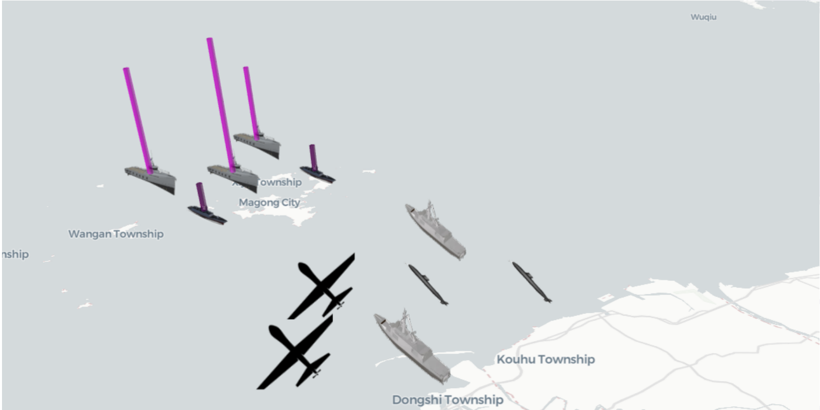
WTA Formulation

This formulation is derived from [Hendrickson et al., 2023]

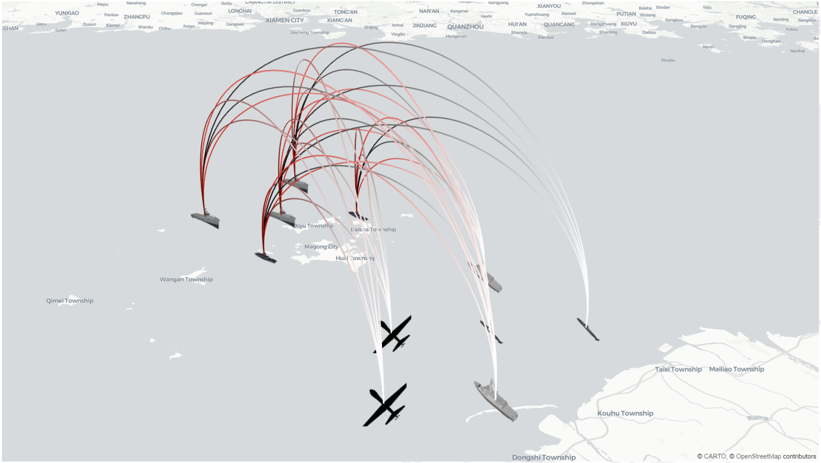
- Indices and Sets
  $i \in 1 \ldots n$ weapons
  $j \in 1 \ldots m$ targets
- Parameters
  V_j: the value of target j
  Pk_{ij}: the probability that weapon i destroys target j
- Decision Variables $x_{ij} \in \{0, 1\}$: whether weapon i is employed against target j

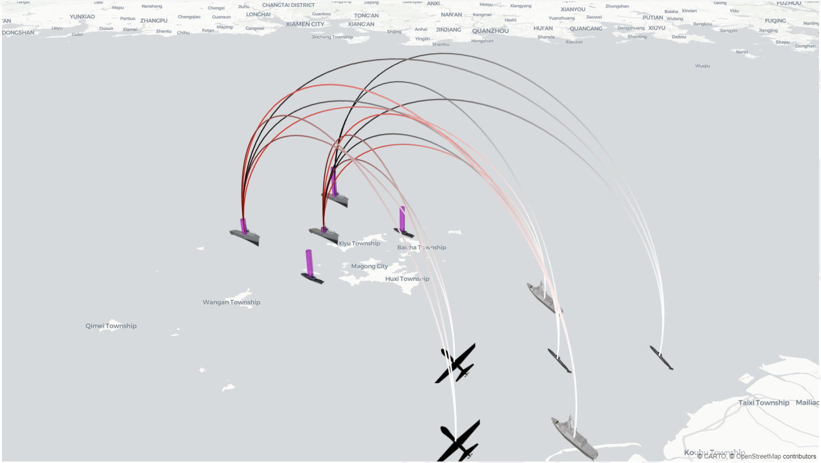
$$\min_x \sum_{j=1}^{m} V_j \prod_{i=1}^{n} (1 - Pk_{ij})^{x_{ij}} \tag{1}$$

$$\text{s.t.} \sum_{j=1}^{m} x_{ij} \leq 1 \quad \forall i \in 1 \ldots n \tag{2}$$


Wuqiu
Township
Magong City
Wangan Township
nship
Kouhu Township
Dongshi Township

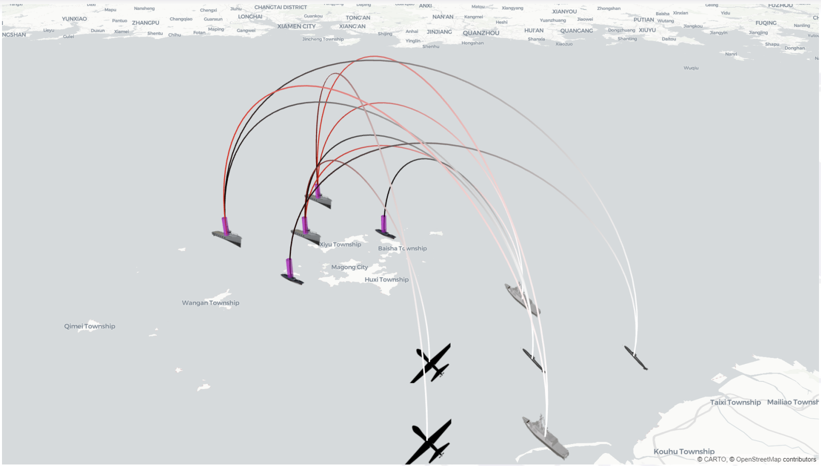
WTA: Toy problem





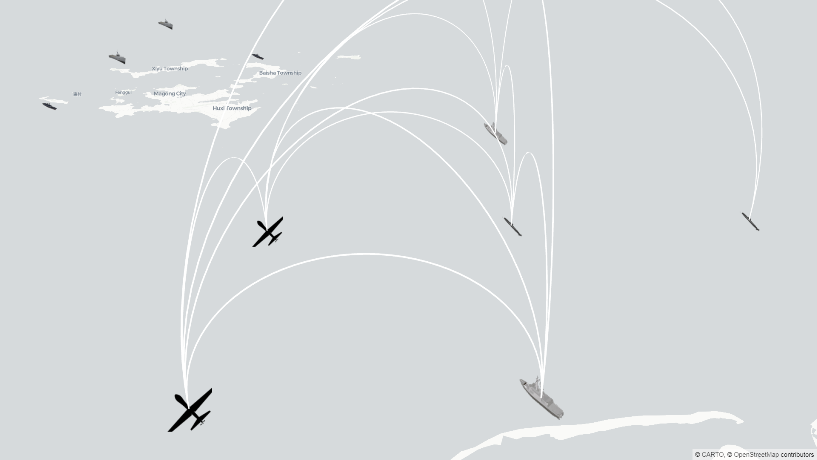
If platform solves their employment independently remaining enemy value is 41.2

WTA: Full solution



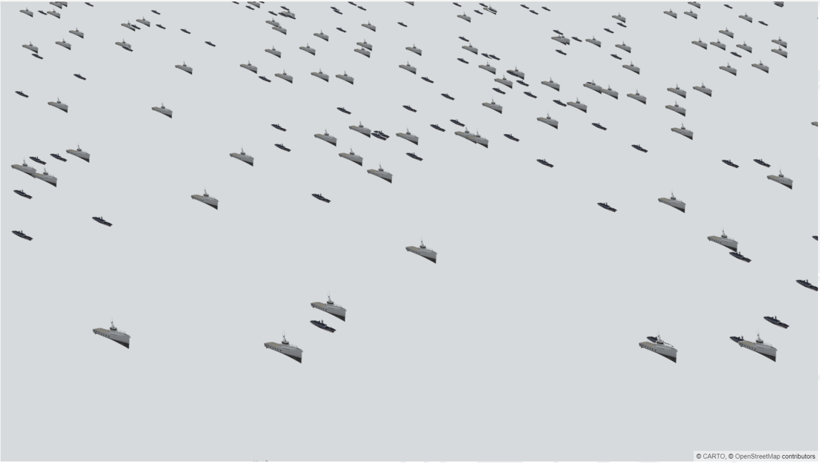
The coordinated decision is much better: remaining enemy value is 33.7

WTA: Feasible comms



Antenna height, radio frequency restrictions, and radiated power can reduce communications options for coordination

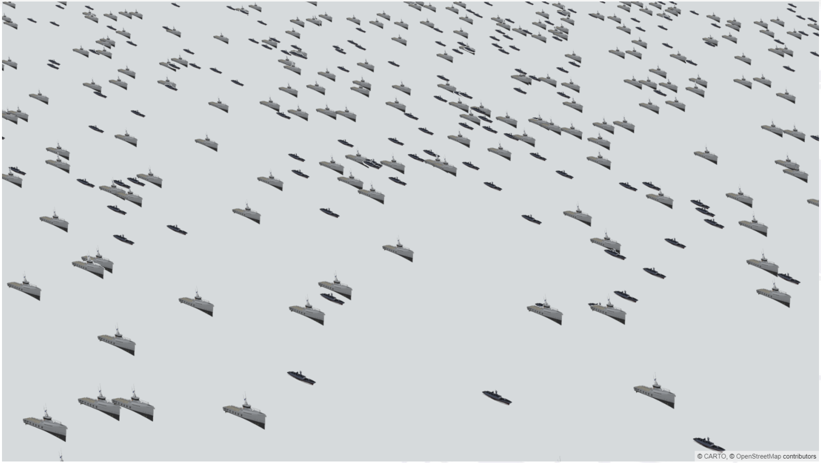
WTA: Large-scale



Why do we need splitting?

large problems

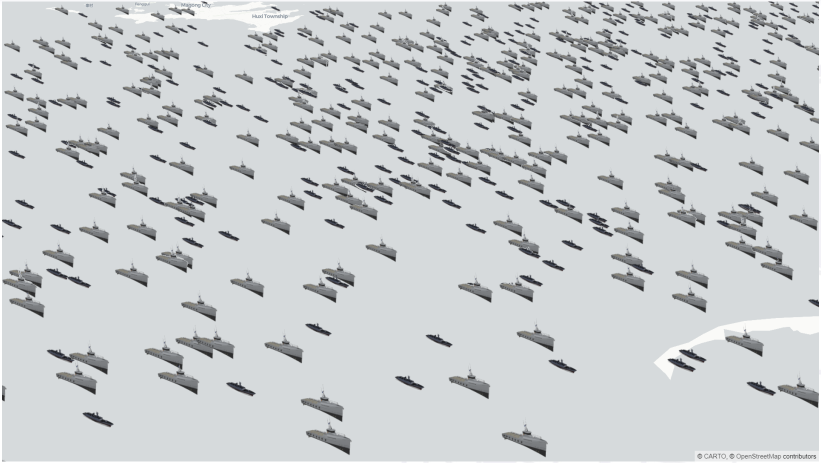
WTA: Large-scale



Why do we need splitting?

very large problems

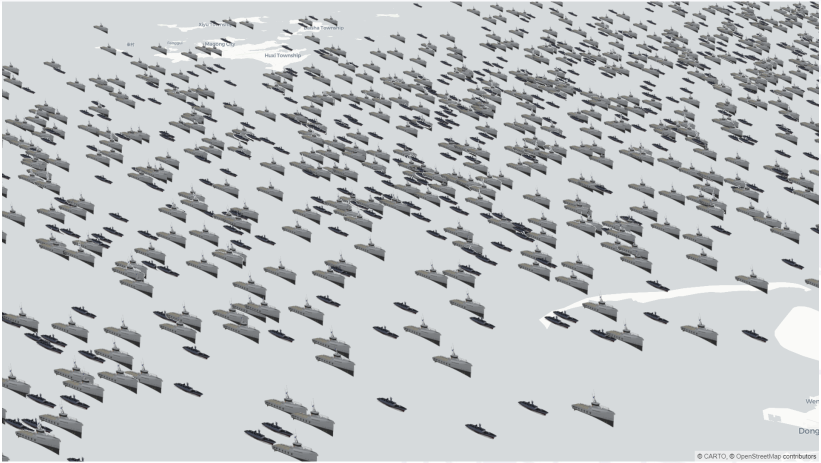
WTA: Large-scale



Why do we need splitting?

Very, very large problems

WTA: Large-scale



Why do we need splitting?

Very, very large problems

Prox Definition

- The prox of a function f at x is defined as $\text{prox}_f(x) = \arg\min_u f(u) + \frac{1}{2}||u - x||^2$.

Prox Definition

- The prox of a function f at x is defined as $\text{prox}_f(x) = \arg\min_u f(u) + \frac{1}{2}||u - x||^2$.

- $z = \arg\min_u f(u) + \frac{1}{2}||u - x||^2$
- $\implies \frac{\partial}{\partial z}\left(f(x) + \frac{1}{2}||z - x||^2\right) \ni 0$
- $\implies 0 \in \partial f(z) + (z - x)$
- $\implies x \in z + \partial f(z)$
- $\implies x \in (\text{Id} + \partial f)(z)$
- $\implies z = (\text{Id} + \partial f)^{-1}(x)$

Resolvent Definition

[Bauschke et al., 2011]

- The resolvent of a (potentially set-valued) operator A, on some $x \in \mathcal{H}$, is given by $J_A(x) = (\mathrm{Id} + A)^{-1}(x)$ where Id is the identity operator.
- Equivalently, if $J_A(x) = z$, we have $x \in (\mathrm{Id} + A)(z)$ or $x \in z + A(z)$.
- Finding a stationary point $x = J_A(x)$ is therefore equivalent to finding a zero of A.
- The resolvent of the subdifferential of a convex, closed, and proper (ccp) function is the prox operator of that function.

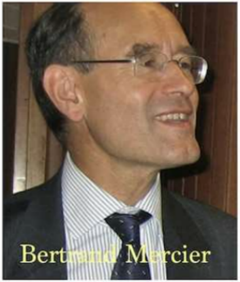
Douglas Rachford:

$$\min_{x \in \mathbb{R}^n} f(x) + g(x)$$

1. $x_1 = J_{\partial f}(z^k) = \mathrm{prox}_f(z^k)$
2. $x_2 = J_{\partial g}(2x_1 - z^k) = \mathrm{prox}_g(2x_1 - z^k)$
3. $z^{k+1} = z^k + x_2 - x_1$

Reformulation as a fixed point operator

$$T(z) = z + J_{\partial g}(2J_{\partial f}(z) - z) - J_{\partial f}(z)$$
$$z^{k+1} = T(z^k)$$



Ryu's Algorithm

- Can Douglas Rachford be extended?
- Frugal resolvent splitting with (possible) lifting
  1. Resolvent Splitting: use only scalar multiplication, addition, and the resolvent
  2. Frugal: evaluate each resolvent only once per iteration
  3. Lifting: increasing the dimension of the fixed point iterates
- For three operators – it can be extended, but only with lifting [Ryu, 2020]

Ryu's Algorithm

$$\min_{x\in\mathcal{H}} f_1(x) + f_2(x) + f_3(x)$$

$$\theta \in (0,1)$$
$$\alpha > 0$$
$$A = \partial f_1$$
$$B = \partial f_2$$
$$C = \partial f_3$$

$$\begin{aligned} x_1 &= J_{\alpha A}(z_1^k) \\ x_2 &= J_{\alpha B}(x_1 + z_2^k) \\ x_3 &= J_{\alpha C}(x_1 - z_1^k + x_2 - z_2^k) \end{aligned}$$

$$\begin{pmatrix} z_1 \\ z_2 \end{pmatrix}^{k+1} = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}^{k} + \theta \begin{pmatrix} x_3 - x_1 \\ x_3 - x_2 \end{pmatrix}$$

Malitsky and Tam Algorithm [Malitsky and Tam, 2023]

- Build frugal resolvent splitting algorithms for any n subdifferentials of convex functions
- Show that the minimal lifting is $n-1$.

$$\min_{x\in\mathcal{H}} \sum_{i=1}^{n} f_i(x)$$

$$\begin{aligned} x_1 &= J_{A_1}(z_1^k) \\ &\vdots \\ x_i &= J_{A_i}(x_{i-1} + z_i^k - z_{i-1}^k) \\ &\vdots \\ x_n &= J_{A_n}(x_1 + x_{n-1} - z_{n-1}^k) \end{aligned}$$

$$\begin{pmatrix} z_1 \\ \vdots \\ z_i \\ \vdots \\ z_{n-1} \end{pmatrix}^{k+1} = \begin{pmatrix} z_1 \\ \vdots \\ z_i \\ \vdots \\ z_{n-1} \end{pmatrix}^{k} + \gamma \begin{pmatrix} x_2 - x_1 \\ \vdots \\ x_{i+1} - x_i \\ \vdots \\ x_n - x_{n-1} \end{pmatrix}$$

- Malitsky and Tam recognize that these algorithms are defined by two coefficient matrices (for x and z)

Coefficient-based definition

For $x \in \mathcal{H}$, $W \in \mathbb{R}^{p \times n}$, and identity Id, let
$$\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{H}^n, \quad \mathbf{W} = W \otimes \mathrm{Id}$$

Then for $\mathbf{z} \in \mathcal{H}^d$, $\mathbf{x} \in \mathcal{H}^n$, any frugal resolvent splitting can be written as:

$$T(\mathbf{z}) := \mathbf{z} + \gamma \mathbf{M}\mathbf{x}, \quad \mathbf{y} = \mathbf{B}\mathbf{z} + \mathbf{L}\mathbf{x}, \quad \mathbf{x} = J_{\mathbf{F}}(\mathbf{y})$$

where $\mathbf{F} = (F_1, \ldots, F_n)$, and $M \in \mathbb{R}^{d \times n}$, $B \in \mathbb{R}^{n \times d}$, $L \in \mathbb{R}^{n \times n}$ are coefficient matrices for $\mathbf{M}, \mathbf{B}$, and $\mathbf{L}$.

Frugal Resolvent Splitting Assumptions

1. $\ker M = \operatorname{span}(\{\mathbb{1}_n\})$
2. $B = -M^T$
3. $L + L^T - 2\mathrm{Id} + M^T M \preceq 0$
4. $\sum_{i,j} L_{ij} = n$ and L is lower triangular
5. For every $\mathbf{z} \in \mathcal{H}^d$, there is a unique $\bar{x} \in \mathcal{H}$ such that for $\mathbf{x} = \mathbb{1} \otimes \bar{x}$
$$\mathbf{x} = J_{\mathbf{F}}(\mathbf{Bz} + \mathbf{Lx})$$
For this assumption, it suffices to have one row of L sum to zero.
6. $\|L\| < 1$

Douglas Rachford Revisited

$$T(z) = z + \gamma(x_2 - x_1)$$

$$x_1 = J_{F_1}(z)$$
$$x_2 = J_{F_2}(2x_1 - z)$$

$$M = -B^T = [-1, 1]$$

$$L = \begin{bmatrix} 0 & 0 \\ 2 & 0 \end{bmatrix}$$

Ryu Revisited

$$T(\mathbf{z}) = \mathbf{z} + \gamma \begin{pmatrix} x_3 - x_1 \\ x_3 - x_2 \end{pmatrix}$$

$$x_1 = J_{F_1}(z_1)$$
$$x_2 = J_{F_2}(x_1 + z_2)$$
$$x_3 = J_{F_3}(x_1 + x_2 - z_1 - z_2)$$

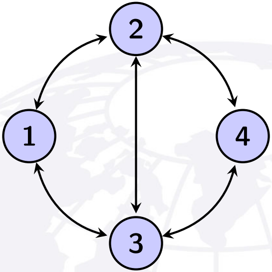
$$M = -B^T = \begin{bmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \end{bmatrix}$$

$$L = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

Malitsky and Tam Revisited

$$T(\mathbf{z}) = \mathbf{z} + \gamma \begin{pmatrix} x_2 - x_1 \\ \vdots \\ x_{i+1} - x_i \\ \vdots \\ x_n - x_{n-1} \end{pmatrix} \quad M = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ 0 & 0 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & -1 & 1 \end{bmatrix}$$

$$\begin{aligned} x_1 &= J_{F_1}(z_1) \\ &\vdots \\ x_i &= J_{F_i}(x_{i-1} + z_i - z_{i-1}) \\ &\vdots \\ x_n &= J_{F_n}(x_1 + x_{n-1} - z_{n-1}) \end{aligned} \quad L = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 0 & 0 & \dots & 1 & 0 \end{bmatrix}$$



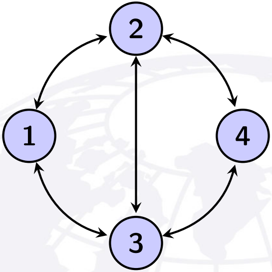
Graph Laplacian

Given diagonal node degree matrix D and adjacency matrix A, graph Laplacian W is:

$$W = D - A$$

Example

$$D = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}, A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}, W = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}$$



Graph Laplacian

If $M \in \mathbb{R}^{E \times V}$ is any directed edge adjacency matrix for the graph, we also have $W = M^T M$

Example

$$M = \begin{bmatrix} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \qquad M^T M = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix} = W$$

Connectivity

- Node connectivity: minimum number of node deletions required to disconnect the graph
- Edge connectivity: minimum number of edge deletions required to disconnect the graph

Algebraic Connectivity

- Based on spectral analysis (eigenvalues) of the graph Laplacian.
- W is positive semi-definite.
- Because all rows of W sum to zero, the smallest eigenvalue $\lambda_1 = 0$.
- The second smallest eigenvalue (or Fiedler value) gives the algebraic connectivity of the graph. [Fiedler, 1973]

Assumptions

1. $\ker M = \operatorname{span}(\{\mathbb{1}_n\})$
2. $B = -M^T$
3. $L + L^T - 2\mathrm{Id} + M^T M \preceq 0$
4. $\sum_{i,j} L_{ij} = n$ and L is lower triangular
5. For every $\mathbf{z} \in \mathcal{H}^d$, there is a unique $\bar{x} \in \mathcal{H}$ such that for $\mathbf{x} = \mathbb{1} \otimes \bar{x}$
$$\mathbf{x} = J_{\mathbf{F}}(\mathbf{B}\mathbf{z} + \mathbf{L}\mathbf{x})$$
For this assumption, it suffices to have one row of L sum to zero.
6. $\|L\| < 1$

SDP formulation

$$
\begin{aligned}
\min_{L,W\in\mathbb{R}^{n\times n}} \quad & \phi(L,W) \\
\text{s.t.} \quad & W\mathbb{1}_n = 0 && \text{(3a)} \\
& L + L^T - 2I + W \preceq 0 && \text{(3b)} \\
& \lambda_1(W) + \lambda_2(W) \geq c && \text{(3c)} \\
& \sum L_{ij} = n && \text{(3d)} \\
& W \succeq 0 && \text{(3e)} \\
& W \in \mathcal{W} && \text{(3f)} \\
& L \in \mathcal{L} && \text{(3g)}
\end{aligned}
$$

This problem is convex!

Find M (weighted edge adjacency matrix)

- W satisfies the requirements to be a graph Laplacian
- e nonzero entries in lower triangle of W correspond to edges
- Define $M \in \mathbb{R}^{e \times n}$
- Walk through nonzero entries in lower triangle of W, creating entries in sequential rows for each edge.

Find B

- $B = -M^T$

Subproblems

- Once the algorithm has been designed, each node sequentially solves the subproblem below.
- x_i is the copy of the decision variables being optimized by node i.
- If all required previous nodes have provided solutions, computation can run in parallel with other nodes.

$$\min_{x_i} f_i(x_i) + \frac{1}{2}\left\|(1 - L_{ii})x_i - \sum_{j=1}^{i-1} L_{ij}x_j^k - \sum_{l=1}^{d} B_{il}z_l^k\right\|_2^2 \tag{4}$$

$$z_l^{k+1} = z_l^k + \gamma \left(\mathbf{M}\mathbf{x}^k\right)_l$$

Subproblems

- Once the algorithm has been designed, each node sequentially solves the subproblem below.
- x_i is the copy of the decision variables being optimized by node i.
- If all required previous nodes have provided solutions, computation can run in parallel with other nodes.

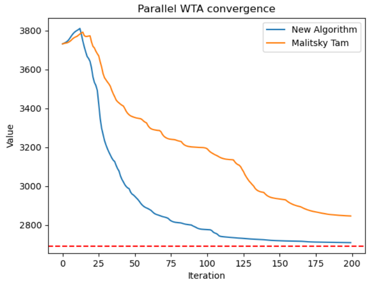
$$\min_{x_i} f_i(x_i) + \frac{1}{2}\left\|(1 - L_{ii})x_i - \sum_{j=1}^{i-1} L_{ij}x_j^k - v_i^k\right\|_2^2 \tag{4}$$

$$v_i^{k+1} = v_i^k - \gamma\left(\mathbf{W}\mathbf{x}^k\right)_i$$

Critical Path Cycle Minimization Algorithm Design Program

$$
\begin{aligned}
\min_{x,s,Z,W} \quad & \max_{k=1}^{n} s_{3n-1,k} \\
\text{s.t.} \quad & W\mathbb{1}_n = 0 && \text{(5a)} \\
& Z - W \succeq 0 && \text{(5b)} \\
& \lambda_1(W) + \lambda_2(W) \geq c && \text{(5c)} \\
& \sum Z_{ij} = 0 && \text{(5d)} \\
& W \in \mathcal{W}, \quad Z \in \mathcal{Z} && \text{(5e)} \\
& s_{ij} - s_{ik} \geq (t_{kj} + m)x_{jk} - m \quad \forall i, k, j > k && \text{(5f)} \\
& s_{i+1,j} - s_{ik} \geq (t_{kj} + m)x_{jk} - m \quad \forall i, k, j \neq k && \text{(5g)} \\
& n x_{kj} \geq |Z_{jk}| \quad \forall j > k && \text{(5h)} \\
& n x_{kj} \geq |W_{kj}| \quad \forall j < k && \text{(5i)}
\end{aligned}
$$

Results

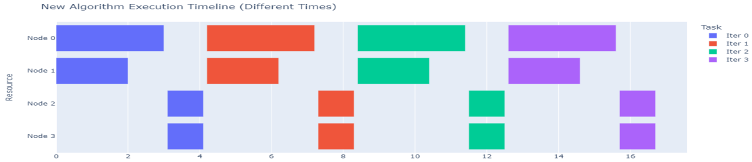


- Built algorithms for different node calculation times
- Current implementation allows quick calculation of up to 4 nodes
- One algorithm performs better than Malitsky Tam!

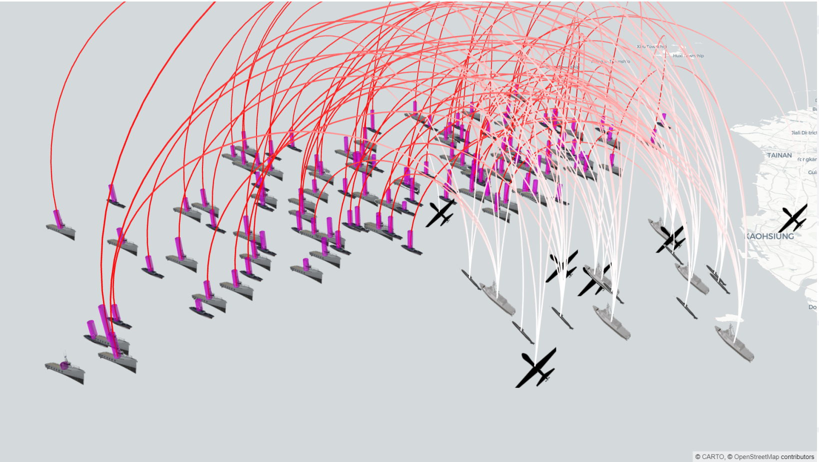
$$L: \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1.5 & 0.5 & 0 & 0 \\ 0.5 & 1.5 & 0 & 0 \end{pmatrix}$$

$$W: \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 2 & -0.5 & -1.5 \\ -1 & -0.5 & 1.67 & -0.17 \\ 0 & -1.5 & -0.17 & 1.67 \end{pmatrix}$$

Algorithm Execution



Results



Future Work

1. Exploration of the convergence rate of various network structures
2. Determination of the impact on convergence of the ordering of the functions in the finite sum.
3. Evaluation of various algorithm generation SDP objective functions and constraints
4. Application to additional optimization problems

Implementation

- Using cvxpy for non-linear convex optimization [Diamond and Boyd, 2016]
  - Provides interface for both SDP and subproblems
- Using YALMIP in MATLAB for Mixed Integer SDP solutions. [Löfberg, 2004]

Questions?

SDP Code

```python
# assumes n, penalty matrices PW, PL
W = cvx.Variable((n,n), PSD=True)
Z = cvx.Variable((n,n), PSD=True)

cons = [Z - W >> 0,
        cvx.lambda_sum_smallest(W, 2) >= connectivity,
        cvx.sum(W, axis=1) == 0,
        cvx.sum(Z) == 0]
cons += [Z[i,i] <= 3 for i in range(n)]
cons += [Z[i,i] >= 2 for i in range(n)]

Z_penalty = cvx.sum(cvx.multiply(PL, cvx.abs(Z)))
W_penalty = cvx.sum(cvx.multiply(PW, cvx.abs(W)))
obj = cvx.Minimize(Z_penalty + W_penalty)
prob = cvx.Problem(obj, cons)
prob.solve()
```

References

Bauschke, H. H., Combettes, P. L., et al. (2011).
Convex analysis and monotone operator theory in Hilbert spaces, volume 408.
Springer.

Diamond, S. and Boyd, S. (2016).
CVXPY: A Python-embedded modeling language for convex optimization.
Journal of Machine Learning Research, 17(83):1–5.

Fiedler, M. (1973).
Algebraic connectivity of graphs.
Czechoslovak mathematical journal, 23(2):298–305.

Hendrickson, K., Ganesh, P., Volle, K., Buzaud, P., Brink, K., and Hale, M. (2023).
Decentralized weapon–target assignment under asynchronous communications.
Journal of Guidance, Control, and Dynamics, 46(2):312–324.

Löfberg, J. (2004).
Yalmip : A toolbox for modeling and optimization in matlab.
In *In Proceedings of the CACSD Conference*, Taipei, Taiwan.

Malitsky, Y. and Tam, M. K. (2023).
Resolvent splitting for sums of monotone operators with minimal lifting.
Mathematical Programming, 201(1-2):231–262.

Ryu, E. K. (2020).
Uniqueness of drs as the 2 operator resolvent-splitting and impossibility of 3 operator resolvent-splitting.
Mathematical Programming, 182(1-2):233–273.