Group Meeting - August 20, 2021

Truong Son Hy *

*Department of Computer Science
The University of Chicago

Ryerson Physical Lab

Content

Content:

1. **Optimization-Based Algebraic Multigrid CoarseningUsing Reinforcement Learning**, https://arxiv.org/abs/2106.01854
2. **Weisfeiler and Lehman Go Topological: Message Passing Simplicial Networks** (ICML 2021) https://arxiv.org/abs/2103.03212 (no slides)

Paper 1

Optimization-Based Algebraic Multigrid CoarseningUsing Reinforcement Learning

https://arxiv.org/abs/2106.01854

Algebraic Multigrid (1)

Introduction

- AMG algorithms aim to solve a sparse linear system of the form $Ax = b$ by successively constructing coarser representations of the problem.
- Constructing an AMG method is effectively a **graph coarsening problem**. Optimal partitioning into coarse and fine nodes is known to be **NP-hard**.

Notation:

- $P \in \mathbb{R}^{n \times n_c}$: Interpolation operator.
- $A_c = P^T A P \in \mathbb{R}^{n_c \times n_c}$: Coarse-grid operator by Galerkin product.

Algorithm 1 Two-Level AMG Algorithm

1: **Input**: Sparse matrix $A \in \mathbb{R}^{n \times n}$, right-hand side $b \in \mathbb{R}^n$, initial guess $x^{(0)} \in \mathbb{R}^n$, interpolation matrix $P \in \mathbb{R}^{n \times n_c}$, coarse-grid matrix $A_c \in \mathbb{R}^{n_c \times n_c}$, convergence tolerance δ, numbers of relaxation sweeps $N_1, N_2 \in \mathbb{N}$, and $k = 0$.
2: **repeat:**
3: Perform N_1 pre-relaxation sweeps on $x^{(k)}$ to obtain $\tilde{x}^{(k)}$
4: Calculate the residual: $\tilde{r}^{(k)} = b - A\tilde{x}^{(k)}$
5: Project the residual to the coarse grid and solve: $A_c e_c^{(k)} = P^T \tilde{r}^{(k)}$.
6: Interpolate and add the coarse-grid correction: $\hat{x}^{(k)} = \tilde{x}^{(k)} + P e_c^{(k)}$
7: Perform N_2 post-relaxation sweeps on $\hat{x}^{(k)}$ to get $x^{(k+1)}$
8: $k = k + 1$, compute $r^{(k+1)} = b - A x^{(k+1)}$
9: **until:** $||r^{(k+1)}|| < \delta$

Algebraic Multigrid (3)

Define a binary variable indicating if a node is fine or coarse:

$$f_i = \begin{cases} 1 & \text{if } i \in F, \\ 0 & \text{if } i \in C, \end{cases}$$

where F and C denote the sets of fine and coarse nodes, respectively.
Optimization:

$$\max \sum_{i=1}^{n} f_i$$
$$\text{s.t. } |a_{ii}| \geq \theta \sum_{j \in F} |a_{ij}|, \; \forall i : f_i = 1.$$

Here, θ is the dominance parameter (in practice, $\theta = 0.56$). This is an **NP-hard** problem. Usually solved by the simulated annealing (SA) approach or greedy algorithms.

Proposal

Proposal

- **Reinforcement Learning:** Dueling Double DQN (a variant of Deep Q-Learning) with an MPNN as the RL agent.
- **Goal:** Computational feasibility, fast and high-quality fine-coarse partitioning comparing to the SA method.
- **Particular problem to address:** Discretized 2D Poisson equation

$$-\Delta\phi = f$$

where Δ is the Laplace operator and $f(x, y)$ and $\phi(x, y)$ are real-valued functions.

Reinforcement Learning (1)

Define a graph $G = (V, E)$:

- Node $i \in V$ for each variable index i.
- Edge $(i, j) \in E$ if $A_{ij} \neq 0$ for $i \neq j$.
- Binary variable $f_i \in \{0, 1\}$ indicates whether node i is a fine node.
- Binary variable $v_i \in \{0, 1\}$ indicates whether node i violates the diagonal dominance constraint.
- $F = \{i | f_i = 1\}$ is the set of fine nodes.
- $C = \{i | f_i = 0\}$ is the set of coarse nodes.

Reinforcement Learning (2)

- **State space:** $\mathcal{S} = \{(f_i, v_i) | i = 1, .., n\}$ which is $2n$ binary variables.
- **Initial state:** s_0 consists of all fine nodes such that $f_i = 1, \forall i$; while v_i is determined by

$$v_i = \begin{cases} 1 & \text{if } f_i = 1 \text{ and } |a_{ii}| < \theta \sum_{j \in F} |a_{ij}|, \\ 0 & \text{otherwise.} \end{cases}$$

- **Action space:** $\mathcal{A} = \{i | v_i = 1\}$. At each time step, the RL agent chooses one violating fine nodes to convert into a coarse node, and we need to recompute all v after.
- **Reward:** $r(s) = -|C|$, we want to minimize the number of coarse nodes.
- **Terminating condition:** when there are no more actions to take (no nodes are violating).

Reinforcement Learning (3)

Two phases:

1. **Training**: we will learn the state value $V(s)$ and the state action advantage $A(s, a)$ for **a certain size** of graph.
2. **Evaluation:** then we evaluate the agent on large graphs where $V(s)$ is no longer accurate but $A(s, a)$ will continue to be correct.

We only need the output of $A(s, a)$ for evaluation, this provides a scale-invariant solution to the RL problem.

Note: This is similar to what we are doing for wavelet MMF.

Reinforcement Learning (4)

Algorithm 2 Evaluation Algorithm

```
Use Lloyd aggregation to decompose the node set into subdomains {V_1, V_2, ..., V_K}
while constraint (4b) is not satisfied do
    Evaluate the advantage TAGCN network to obtain the advantage A_i for each node
    for k = 1 to K such that V_k contains at least one node with v_i = 1 do
        i = argmax_{j ∈ V_k, v_j = 1} A_j
        Coarsen node i
    end for
end while
```

- Lloyd aggregation is basically a graph decomposition/clustering algorithm.
- Constraint 4b is the diagonal dominance constraint.
- TAGCN is the RL agent (after training).

Experiments (1)

- **Metric:** F-fraction (higher is better) $= |F|/(|F| + |C|)$.
- **Baseline:** Simple greedy method of MacLachlan and Saad.

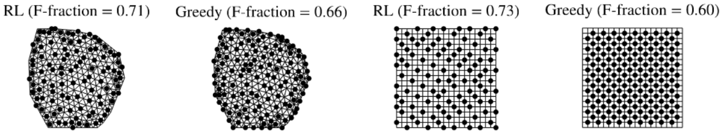
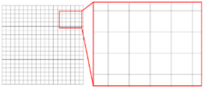
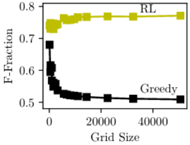
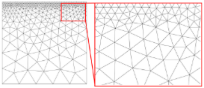
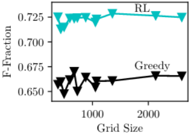
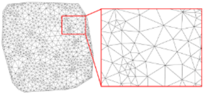
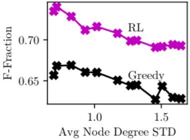


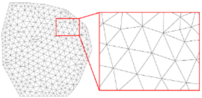
Figure 1: Example coarsenings of meshes from the "Different Size" (left) and "Structured" (right) families, comparing RL and greedy coarsening algorithms.

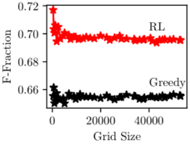
Experiments (2)

<table>
<tr><th>Family</th><th>Example</th><th>F-Fraction vs. Attribute</th></tr>
<tr><td>Structured: A family of 18 rectangular structured grids with different grid sizes.</td><td></td><td>
</td></tr>
<tr><td>Graded Mesh: A family of 12 unstructured grids with different convex shapes, and graded meshes, i.e., smoothly varying mesh size across the domain.</td><td></td><td>
</td></tr>
<tr><td>Wide Valence: A family of 12 unstructured convex grids with the same size and different average node degree standard deviation.</td><td></td><td>
</td></tr>
</table>

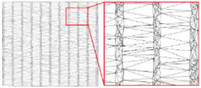
Experiments (3)

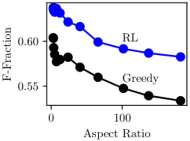
Different Size: A family of 42 unstructured convex grids with grid size ranging from about 60 to 52 000 nodes. The average node degree standard deviation and mesh aspect ratio is constant.



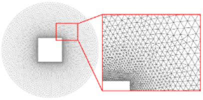


Aspect Ratio: A family of 12 unstructured convex grids with the same size and different average mesh aspect ratio, varying from about 2 to 180.





Non-Convex: A family of 12 unstructured non-convex grids. The grids are generated by removing geometrical shapes from a main geometry and meshing the remainder.



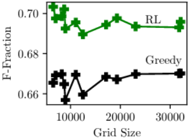


Table 1: Mesh families used for numerical experiments, showing F-fractions (higher is better).

Experiments (4)

Theorem 4. *For the Poisson problem (2) discretized with a 5-point finite difference stencil on a structured grid of size $n_x \times n_y$, the constraint (4b) implies that the F-fraction, f, is bounded by:*

$$f \leq 0.8 + \frac{2}{n_x} + \frac{2}{n_y} - \frac{4}{n_x n_y}. \tag{6}$$

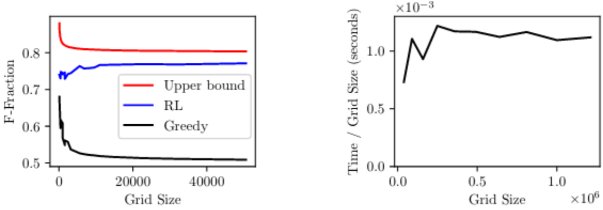


Figure 2: Left: F-fraction for the RL method and comparison methods (higher is better). Right: Evaluation time divided by grid size, showing linear scaling in grid size.