

A Combinatorial Scheme for Developing Efficient Composite Solvers

Sanjukta Bhowmick, Padma Raghavan and Keita Teranishi

E-mail: {bhowmick, raghavan, teranish}@cse.psu.edu

The Pennsylvania State University

Funded by NSF and DOE

Introduction

- Many scientific computing problems (for example, sparse linear system solution) have several competing solution methods
- The best method may be domain or problem dependent and could vary even within across time-steps of the same simulation
- Each method can potentially represent a trade-off between a performance metric and robustness metric
- A composite would consist of several methods in a sequence; failure of one method would result in the execution of the next method in the sequence

Composite Methods

- A composite is significantly more robust than its component methods
- The robustness of a composite is independent of the order in which methods are selected
- The performance of the composite depends on the order in which methods are selected
- Our goal is to develop composites with optimal performance

A Combinatorial Framework-I

- There are n methods M_1, M_2, \dots, M_n ; M_i is associated with two measures:
 - normalized execution time: t_i (performance metric)
 - success rate: r_i (robustness metric); failure rate $f_i = 1 - r_i$
- The set \mathbf{P} contains all permutations (of length n) of $\{1, 2, \dots, n\}$
- \hat{P}_i denotes the i -th method in $\hat{P} \in \mathbf{P}$; the composite \hat{C} is specified by \hat{P}
- \hat{C} executes methods in the order specified by \hat{P} ; partial results of a failed method are not reused

A Combinatorial Framework-II

- Reliability of \hat{C} : $1 - \prod_{i=1}^n f_i$; it does not depend on \hat{P}
- Execution time (worst case) of \hat{C} :

$$\hat{T} = t_{\hat{P}_1} + f_{\hat{P}_1} t_{\hat{P}_2} + \cdots + f_{\hat{P}_1} f_{\hat{P}_2} \cdots f_{\hat{P}_{n-1}} t_{\hat{P}_n}$$

- The performance of a composite depends directly on the permutation and there can be dramatic variations
- Problem: Determine $\tilde{P} \in \mathbf{P}$ such that the associated composite \tilde{C} is optimal, i.e., its worst case time $\tilde{T} = \min\{\hat{T} : \hat{P} \in \mathbf{P}\}$

Composites of 3 Methods: An Example

t_1, r_1, f_1

1, 0.1, 0.9

t_2, r_2, f_2

1.5, 0.7, 0.3

t_3, r_3, f_3

3.0, 0.8, 0.2

Permutation

Composite Time

1,2,3 'least time'

$$3.16 = 1 + .9 \times 1.5 + .9 \times .3 \times 3.0$$

3,2,1 'least failure'

$$3.36 = 3 + .2 \times 1.5 + .2 \times .3 \times 1.0$$

2,1,3

$$2.61 = 1.5 + .3 \times 1 + .3 \times .9 \times 3.0$$

2,3,1

$$2.46 = 1.5 + .3 \times 3 + .3 \times .2 \times 1.0$$

Failure rate of any composite is 0.054

And The Optimal Composite is

- Define $u_i = \frac{t_i}{r_i}$ as the utility ratio of method M_i
- Let $\tilde{P} \in \mathbf{P}$ and let \tilde{C} be the associated composite
- Theorem: \tilde{C} is the optimal composite with $\tilde{T} = \min\{\hat{T} : \hat{P} \in \mathbf{P}\}$ if and only if $\tilde{P} = \tilde{P}_1, \tilde{P}_2, \dots, \tilde{P}_n$ is such that

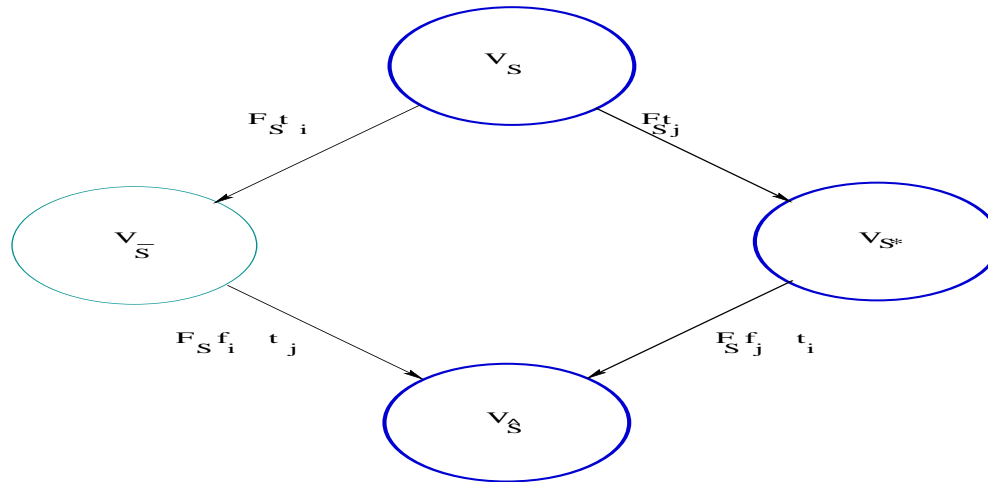
$$u_{\tilde{P}_1} \leq u_{\tilde{P}_2} \leq \dots \leq u_{\tilde{P}_{n-1}} \leq u_{\tilde{P}_n}$$

- The optimal composite is one in which the component methods are arranged in increasing order of their utility ratios
- The proof of the if part is tedious, only-if part is more intuitive

Composites as Paths in a Layered Graph

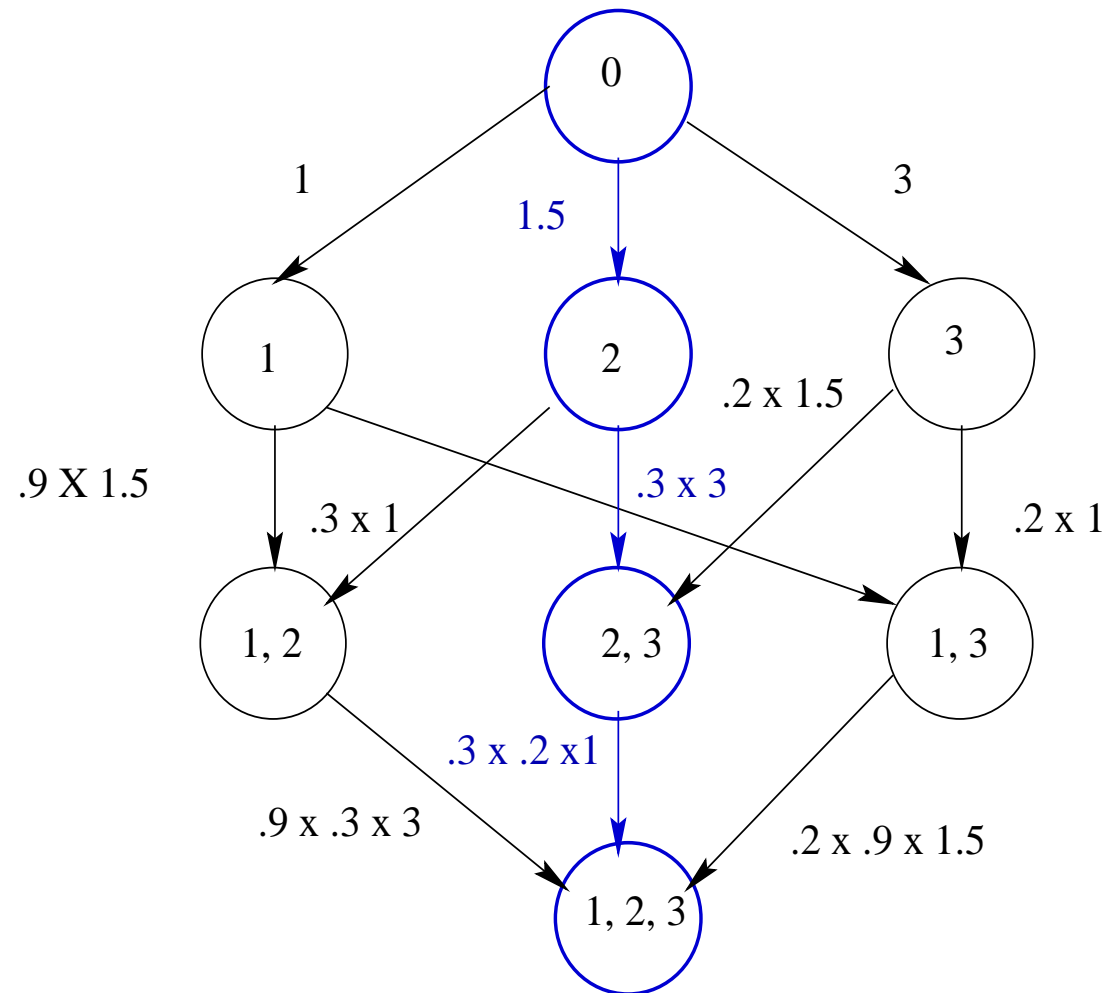
- A graph with vertices at $n + 1$ levels, a vertex at level i represents a subset of i methods (V_0 at level 0 for the empty set)
- An edge connects vertex \bar{S} to \hat{S} if they are on adjacent levels, $|\hat{S} \setminus \bar{S}| = 1$, and $\hat{S} \cap \bar{S} = \bar{S}$
- For the subset of methods S , $F_S = \prod_{M_i \in S} f_i$ and $F_\Phi = 1$; edge $\bar{S} \rightarrow \hat{S}$ has weight $F_{\bar{S}} t_i$ if $\hat{S} \setminus \bar{S} = \{i\}$
- A path from V_Φ to $V_{\{1,2,\dots,n\}}$ denotes a composite in the order in which methods were added to vertex sets
- The length of the path is the time for the composite; the **optimal** composite is the one with the **shortest** path

Composites as Paths in a Layered Graph



- V_S and $V_{\hat{S}}$ are on shortest path and $\hat{S} - S = \{i, j\}$.
- There are only 2 paths from V_S to $V_{\hat{S}}$, one with $V_{\bar{S}}$ ($\bar{S} - S = \{i\}$) and another with V_{S^*} ($S^* - S = \{j\}$)
- By shortest paths, $T_S + F_S t_j + F_S f_j t_i \leq T_S + F_S t_i + F_S f_i t_j$; simplifies to $t_j(1 - f_i) \leq t_i(1 - f_j)$ and thus $u_j \leq u_i$

An Example For 3 Methods



Composites of 3 Methods: An Example

 t_1, r_1, f_1, u_1
 $1, 0.1, 0.9, 10$
 t_2, r_2, f_2, u_2
 $1.5, 0.7, 0.3, 2.14$
 t_3, r_3, f_3, u_3
 $3.0, 0.8, 0.2, 3.75$

Permutation

Composite Time

1,2,3 'least time'

$$3.16 = 1 + .9 \times 1.5 + .9 \times .3 \times 3.0$$

3,2,1 'least failure'

$$3.36 = 3 + .2 \times 1.5 + .2 \times .3 \times 1.0$$

2,1,3

$$2.61 = 1.5 + .3 \times 1 + .3 \times .9 \times 3.0$$

2,3,1

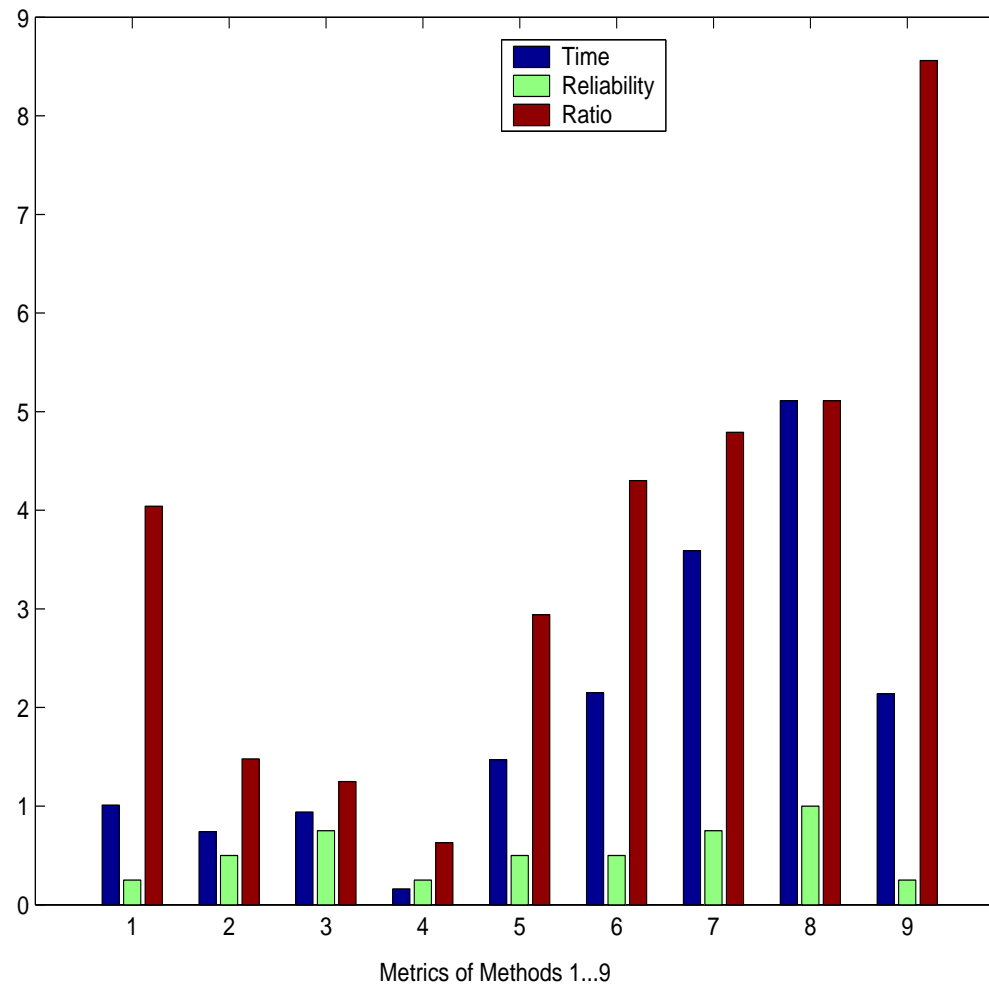
$$2.46 = 1.5 + .3 \times 3 + .3 \times .2 \times 1.0$$

Failure rate of any composite is 0.054

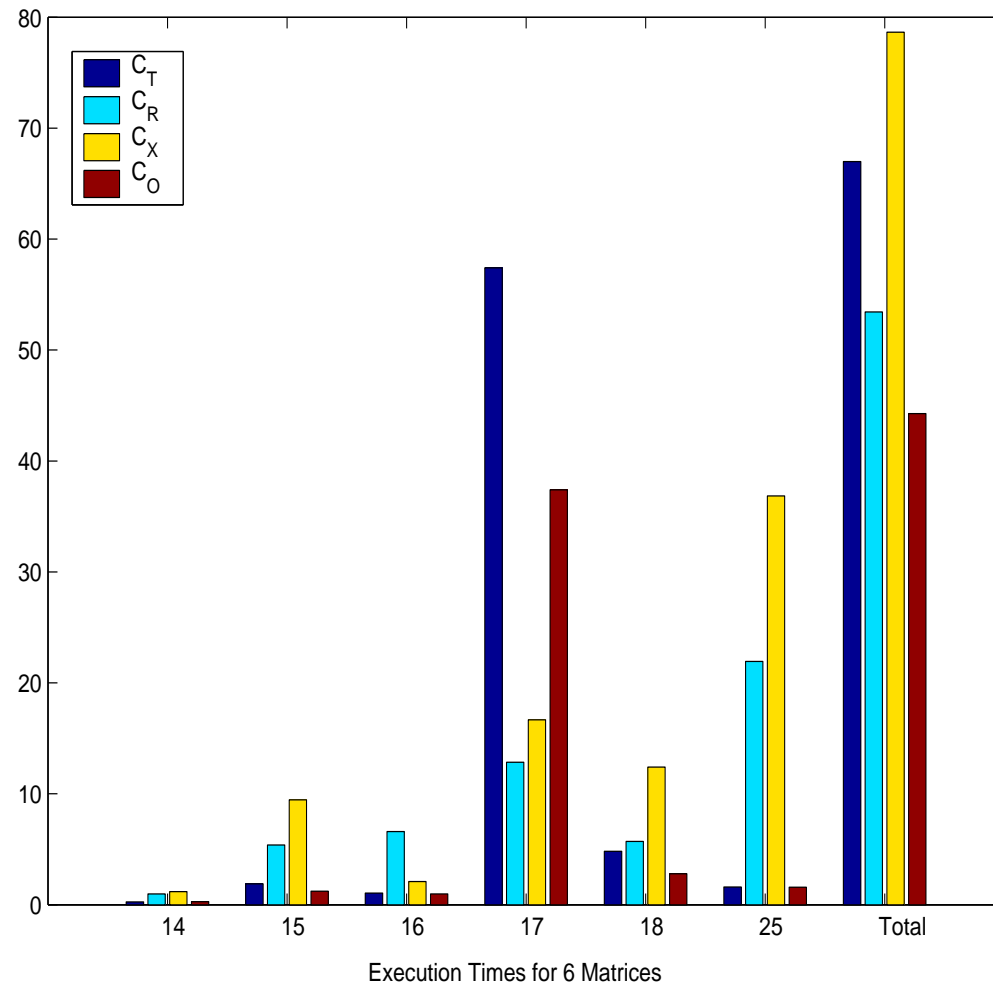
Experiments

- Solve sparse $Ax = b$, A is SPD using 9 methods, CG, preconditioned CG with: SOR, Jacobi, incomplete Cholesky with 0/1/2/3 levels of fill and .0001 to .01 drop thresholds
- Used the geometric mean of the normalized running times on a sample set as t_i , and observed success rates as r_i
- Observed execution times of four composites on larger set:
 - C_T : increasing order of execution time
 - C_R : decreasing order of robustness
 - C_X : random order
 - C_O : increasing order of utility ratio (optimal)

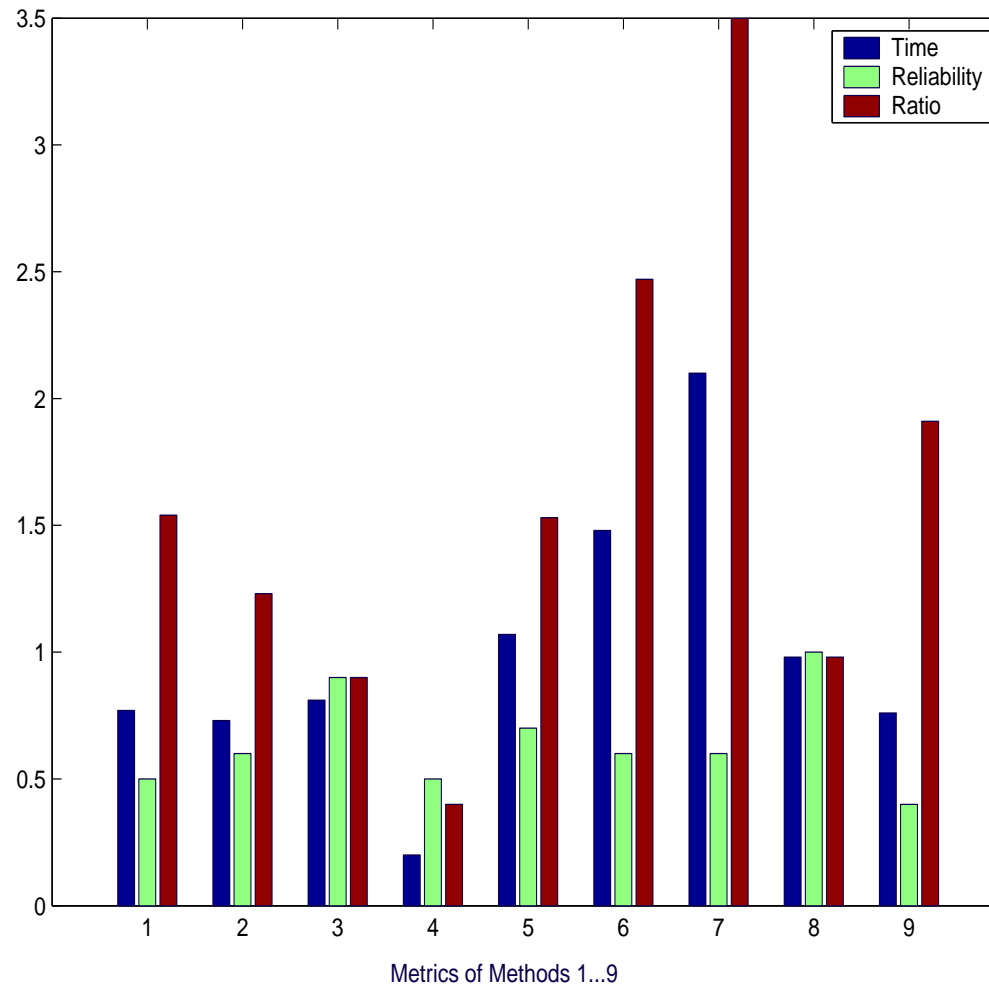
Metrics: bcsstk



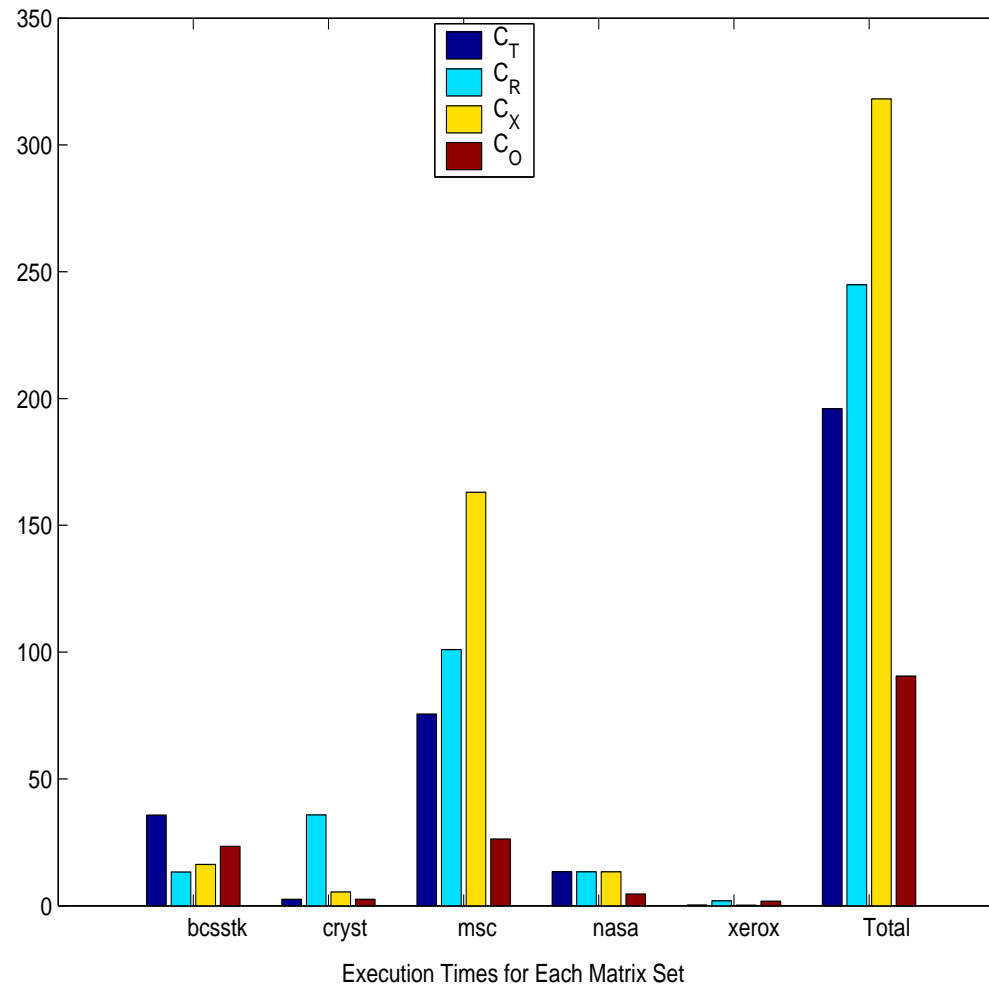
Times:bcsstk



Metrics: bcsstk, cryst, msc, nasa, xerox



Times:bcsstk, cryst, msc, nasa, xerox



Conclusions

- Composite methods are particularly relevant with emerging “component architectures” and “plug-and-play” simulation environments; a solver component could dynamically use observed execution times and failure rates to implement composites
- Related problems:
 - incorporating partial reuse of results from a failed method
 - determining statistically valid performance and reliability metrics by sampling
 - generalization to a parallel computing model