Proceedings of the 2004 Winter Simulation Conference
R .G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds.

APPROXIMATING COMPONENT SELECTION

Michael Roy Fox
David C. Brogan
Paul F. Reynolds, Jr.

151 Engineer's Way
P.O. Box 400740
Department of Computer Science, University of Virginia
Charlottesville, VA 22904, U.S.A.

ABSTRACT

Simulation composability is a difficult capability to achieve due to the challenges of creating components, selecting combinations of components, and integrating the selected components. We address the second of these challenges through analysis of Component Selection (CS), the NP-complete process of selecting a minimal set of components to satisfy a set of objectives. Due to the high order of computational complexity of CS, we examine approximating solutions that make the CS process practicable. We define two variations of CS and prove that good approximations to optimal solutions result from applying a standard Greedy selection algorithm to each. Despite our creation of approximable variations of CS, we conjecture that any proof of the inapproximability of CS will reveal theoretical limitations of its practicality. We conclude that reasonably constrained variations of CS can be solved satisfactorily, and efficiently, but more general cases appear to never be solvable in a similar manner.

1 INTRODUCTION

Composability is the ability to combine reusable simulation components to satisfy a set of user objectives. Composability is a highly sought-after goal for model and simulation developers because of the benefits afforded by reuse. Component Selection (CS) is an NP-complete optimization problem formally shown to be embedded within composability (Petty, Weisel, and Mielke 2003). It was conjectured to be NP-complete in (Page and Opper 1999) and proven to be so in (Petty, Weisel, and Mielke 2003). It is the problem of choosing the minimum number of components from a set of components such that their composition satisfies a set of objectives. For composability to be achievable in reasonable time, the selection of a set of components must be accomplishable in polynomial-time. In this paper we analyze the approximability of CS by evaluating the performance of the standard Greedy approximation algorithm on it and constrained variations of it. We choose to study Greedy on CS because Greedy is one of the best polynomial-time algorithms for approximating Minimum Set Cover (MSC), a closely related NP-complete problem shown to be reducible to CS (Slavík 1996).

We constrain CS by limiting emergence. Informally, emergence occurs when the set of objectives satisfied by a set of components is greater than the sum of the parts: some of the objectives satisfied cannot be satisfied by any of the components alone. In one variation of the problem, we assume emergent behavior exists equally and uniformly amongst the components, and in the other we assume emergent behavior exists only amongst compositions consisting of n or fewer components.

CS is a relatively new complexity problem and has, in the past, only been studied with regard to composability. However, we maintain that it is not only embedded within composability but also within any other simulation/modeling methodology that makes use of simulation/model repositories, libraries, and the like. Examples of methodologies enforcing the desire to reuse, and to combine for the purpose of new functionality, pervade the world of simulation, a few being: using HLA (Dahmann, Calvin, and Weatherley 1999), using modules from the Modeling and Simulation Resource Repository (MSRR) of DMSO (Online: http://www.msrr.dmso.mil/ 2002), and using CORBA-based applications for simulation development (Wang, Schmidt, and O'Ryan 2000). These all have the notion of interoperation and cooperation of simulation/model components that were developed and placed in a repository in one form or another to be later reused. Such repositories imply choice of a combination of components, or pieces of a whole, in order to satisfy end-user objectives. CS is the problem that arises when a simulationist seeks to select an optimal set of components to satisfy a set of requirements.

In the sections that follow we provide necessary background knowledge on approximation theory and a

sketch of the proof of the approximation ratio Greedy exhibits on MSC. We describe CS formally and provide necessary definitions for and formal descriptions of our two newly proposed variations of CS. We then show proofs of Greedy's approximation ratio exhibited on each variation, one of which we limit to a special case. Lastly, we provide a concise description of the theory behind the inapproximability of hard problems and present a conjecture that CS, because of the unpredictability of its nature of emergence, is practically inapproximable.

2 APPROXIMATION THEORY

When assessing the practical application of NP-hard or NP-complete problems, it is useful to analyze their approximability, or whether their optimal solutions can be approximated by polynomial-time algorithms (Hromkovic 2001). Approximation theory is concerned with the approximation of optimal solutions to hard optimization problems. Informally, an approximation algorithm is a polynomial-time algorithm for solving an optimization problem that outputs a solution whose size is always within some scaling factor of the optimal solution. Proving the usefulness of the approximation algorithm on a problem means showing that the factor is very close to one and, if in the case of it being a function, grows very slowly with the size of the problem.

Formally, an algorithm for a problem has an *approximation ratio* of $f(n)$, n being the size of the problem, if the cost C_a of the solution produced by the approximation algorithm is within a factor of $f(n)$ of the cost C_o of an optimal solution (Cormen et al. 2001). In the case of a maximization problem, an algorithm is said to be $f(n)$-*approximate* if the ratio C_o to C_a is less than or equal to $f(n)$, whereas in the case of a minimization problem the opposite ratio C_a to C_o is considered.

3 APPROXIMATING MSC WITH GREEDY

We illustrate approximation theory with an example, which we will then use in our proofs concerning the approximability of variations of CS. The problem definition of MSC is as follows (Hromkovic 2001):

> INSTANCE: (X, F) where X is a finite set and F is a set containing subsets of X, such that every element of X belongs to at least one subset in F.
> QUESTION: Is there a subset C of F of size less than or equal to K such that C *covers* X, *or* $X = \cup_{S \in C} S$.

Algorithm 1 illustrates the standard Greedy approximation algorithm on MSC (Hromkovic 2001).

Algorithm 1
Input: (X, F);
Step 1: $C := NULL$;
$U := X$;
Step 2: *while* $U \neq NULL$
do begin choose an $S \in F$ *such that* $|S \cap U|$ *is maximal;*
$U := U - S$;
$C := C \cup \{S\}$;
end;
Output: C;

The following theorem is from (Cormen et al. 2001) and we provide a sketch of its proof. Refer to (Cormen et al. 2001) for the details of the proof, which are unnecessary to include for the purposes of this paper.

Theorem 1 *Algorithm 1 is Har(max{|S| | S ∈ F})-approximate for Minimum Set Cover, where Har(n) is the Harmonic Series ∑(1/k) for k =1 to n.*

Sketch of Proof of Theorem 1 Assign a cost of 1 to each set selected by Algorithm 1. Let $cost_x$ denote the cost allocated to element x, for each $x \in X$. If x is covered for the first time by subset S_i, then

$$cost_x = 1/|S_i - (S_1 \cup S_2 \cup \ldots \cup S_{i-1})| \, . \quad (1)$$

Use Equation (1) to derive the desired relationship between the cost assigned to the optimal set cover C^* and the cost assigned to the set cover C returned by Algorithm (Cormen et al. 2001). The derived relationship is such that

$$|C| \leq |C^*| \cdot Har(\max\{|S| \mid S \in F\}) \, , \quad (2)$$

thus proving the theorem (Cormen et al. 2001).
[End of Sketch of Proof of Theorem 1]

4 APPROXIMATING CS

CS is the problem of choosing the minimum number of components from a set of components such that their composition satisfies a set of objectives. The notation used for formally defining CS, as laid out in (Petty, Weisel, and Mielke 2003) with a few minor changes to improve appearance is: Let $O = \{o_1, o_2, \ldots, o_n\}$ be a set of objectives. Let $C = \{c_1, c_2, \ldots, c_m\}$ be a set of components. A simulation system S is a subset of C, i.e., $S \subseteq C$. If $|S| > 1$ then S is a composition. Let $\circ$ denote composition of components, e.g., $(c_j \circ c_k)$ is a composition of two components. Let $\Rightarrow$ denote satisfying an objective, i.e., $c_j \Rightarrow o_i$ means component c_j satisfies objective o_i, and $c_j \neg\Rightarrow o_i$ means that it does not. The $\Rightarrow$ and $\neg\Rightarrow$ operators may also be applied to compositions and sets of objectives, e.g., $(c_j \circ c_k) \Rightarrow o_i$ and $S \neg\Rightarrow O$ have the expected meanings. A simulation system $S \Rightarrow O$ if and only if $S \Rightarrow o_i$ for every $o_i \in O$.

The set of objectives satisfied by a composition of components is not necessarily the union of the objectives satisfied by the components individually. Table 1 illustrates the rules of the forms of composition: *emergence*, *non-emergence*, and *anti-emergence*. We assume the non-existence of anti-emergence in CS in our approximability analysis because we believe it can be excluded as an attribute by controlling the way in which components are composed, or connected.

The problem definition of CS follows (Petty, Weisel, and Mielke 2003):

> INSTANCE: Set $C = \{c_1, c_2, \ldots, c_m\}$ of components, set $O = \{o_1, o_2, \ldots, o_n\}$ of objectives, oracle function σ: $power(C) \rightarrow power(O)$, positive integer $K \leq |C|$.
> QUESTION: Does C contain a composition S that satisfies O of size K or less, i.e., a subset $S \subseteq C$ with $|S| \leq K$ such that $O \subseteq \sigma(S)$?

Petty, Weisel, and Mielke include the oracle function σ in CS (2003). The theoretical oracle function accepts a set of components as input and, in one step, computes the set of objectives that are satisfied by the composition of the input components. The oracle function subsumes the difficult problem of computing an objective's decidability into one step within the problem of component selection and it allows the examination of component selection to be separated from that of computing an objective's decidability. Petty, Weisel, and Mielke prove CS to be NP-complete by first showing that an optimal solution to it can be verified in polynomial-time and then showing that MSC can be reduced to it in polynomial-time (2003). Since CS is a computational complexity problem associated with composability, CS or variations of CS must be shown to be practically approximable to show or maintain the notion of composability as a practical ability. We examine approximating CS and variations of CS.

4.1 Greedy on CS

In attempting to find or create approximation algorithms for CS, it is intuitive to research approximation algorithms for MSC, because MSC can be reduced to CS in polynomial-time. The Greedy algorithm is the most widely studied approximation algorithm used for MSC. Refer to (Slavík 1996) for a comprehensive study on approximating MSC with Greedy. Our analysis indicates that the Greedy algorithm fails to exhibit a good approximation ratio when applied to the CS problem. We demonstrate this poor performance with an implementation of the Greedy algorithm (Algorithm 2) and an instance of CS to which it is applied (Instance 1).

Algorithm 2
Input: *(C, O);*
Step 1: *S := NULL;*
Z := O;
Step 2: *while Z ≠ NULL*
do begin choose c in C such that
$|\sigma(S \cup \{c\}) \cap Z|$ is maximal;
S := S ∪{c};
Z := Z - σ(S);
end;
Output: *S;*

Instance 1: Let O = {a, b, c, d, e, f, g, h, i} be the set of objectives. Let C = $\{c_1, c_2, \ldots, c_{10}\}$ be the set of components. $c_1 \Rightarrow$ {a, b}, $c_2 \Rightarrow$ {b, c} $c_3 \Rightarrow$ {c, d}… , $c_8 \Rightarrow$ {h,i}, $c_9 \Rightarrow$ {a}, $c_{10} \Rightarrow$ {a}, $(c_9 \circ c_{10}) \Rightarrow$ O.

In Instance 1, the optimal solution is obviously the set containing c_9 and c_{10}. Algorithm 2 would select c_1 through c_8. It selects $|C|$ components instead of the optimal solution containing 2 components, obviously an infeasible solution for similar instances where $|C|$ is large.

4.2 Constraining CS with Emergence Assumptions

As we have shown earlier, Greedy does not perform as well on CS as it does on MSC because of the existence of emergence in CS. The next step is to constrain the emergence within CS to the extent that Greedy solves the constrained version reasonably well. We create two

Table 1: Forms of Composition (Petty, Weisel, and Mielke 2003)

$c_j \Rightarrow o_i$	$c_k \Rightarrow o_i$	$(c_j \circ c_k) \Rightarrow o_i$	Non-emergent
$c_j \neg\Rightarrow o_i$	$c_k \Rightarrow o_i$	$(c_j \circ c_k) \Rightarrow o_i$	Non-emergent
$c_j \Rightarrow o_i$	$c_k \neg\Rightarrow o_i$	$(c_j \circ c_k) \Rightarrow o_i$	Non-emergent
$c_j \neg\Rightarrow o_i$	$c_k \neg\Rightarrow o_i$	$(c_j \circ c_k) \Rightarrow o_i$	Emergent
$c_j \Rightarrow o_i$	$c_k \Rightarrow o_i$	$(c_j \circ c_k) \neg\Rightarrow o_i$	Anti-emergent
$c_j \neg\Rightarrow o_i$	$c_k \Rightarrow o_i$	$(c_j \circ c_k) \neg\Rightarrow o_i$	Anti-emergent
$c_j \Rightarrow o_i$	$c_k \neg\Rightarrow o_i$	$(c_j \circ c_k) \neg\Rightarrow o_i$	Anti-emergent
$c_j \neg\Rightarrow o_i$	$c_k \neg\Rightarrow o_i$	$(c_j \circ c_k) \neg\Rightarrow o_i$	Non-emergent

variations of CS, each constrained by assumptions regarding emergence, and analyze the performance of the Greedy algorithm on each variation separately.

4.2.1 Greedy on *u*-UCS

For the following theorems and proofs we use the term *degree of emergence*, and we define it as the number of objectives satisfied by a composition via emergence.

Definition 1 *A composition S of individual components has a degree of emergence d if and only if* $|\sigma(S)| - |\{\cup_{c \in S} \sigma(c)\}| = d$.

Definition 2 *The set of objectives* $E = \{ \{\sigma(S) - \{\cup_{c \in S} \sigma(c)\} \}$ *is the set containing the emerging objectives of the composition S. If E is not empty, the composition is said to exhibit emergent behavior.*

Definition 3 *An instance of CS exhibits uniform emergence of degree u if and only if the degree of emergence for any pair-wise composition PS of components in the instance is exactly equal to u and the degree of emergence for any composition with cardinality greater than two is equal to zero.*

Definition 4 *An instance of u-Uniform Component Selection (u-UCS) is an instance of CS that exhibits uniform emergence of degree u. Due to the uniform emergence of degree u exhibited by all pairs of a composition S, and the possibility of redundant emergence, the total number of emergent objectives is less than or equal to ((|S| choose 2) · u).*

Theorem 2 *Algorithm 2 is Har(max{|σ(c)| | c ∈ C})-approximate for 0-UCS.*

Proof of Theorem 2 Instances of *0*-UCS exhibit no emergence in any possible composition of components. Therefore, the objectives satisfied by any composition is simply the union of the objectives satisfied by each component in the composition individually. As shown earlier, Greedy is *Har*(max{|*S*| | *S* ∈ *F*}-*approximate* by the derived Equation (2) for MSC. Petty, Weisel, and Mielke showed how MSC can be reduced to CS (2003). In terms of how Greedy applies to each, MSC and *0*-UCS are exactly the same. A subset *S* of *F* in MSC is a subset σ(*c*) of *O* in *0*-UCS, and therefore Algorithm 2 is *Har*(max{|σ(*c*)| | *c* ∈ *C*})-*approximate* for *0*-UCS.

[End of Proof of Theorem 2]

Theorem 3 *Algorithm 2 is Har(max{|σ(c)| | c ∈ C})-approximate for u-UCS, u > 0.*

Proof of Theorem 3 In the worst case, for every component *c* added to *S* in Algorithm 2, the inevitable emergent behavior (defined by *u*) will not cause *S* to satisfy any new objectives (i.e. the emergent objectives had already been satisfied or they were in O). This complete overlapping effect of the worst case causes *u*-UCS, *u* > 0, to be *0*-UCS insofar as the approximation ratio Greedy exhibits on it. The Proof of Theorem 2 now applies and completes the proof of Theorem 3.

[End of Proof of Theorem 3]

Though the theorems for *0*-UCS and *u*-UCS, *u* > 0, are the same, we separate them with the purpose of indicating the potential for *u*-UCS, *u* > 0, to be more quickly solved than *0*-UCS. If we relax the worst-case condition that emergent properties of each added component in Algorithm 2 provide no benefit in satisfying the problem's objectives, (a more typical scenario), the algorithm will more quickly satisfy all the objectives due to emergent behavior at every step. This causes Algorithm 2 to finish more quickly and to potentially output a smaller solution than the *0*-UCS instance.

4.2.2 Greedy on *n*-CS

In this section we analyze a useful variation of CS we call *n*-CS.

Definition 5 *Instances of n-CS are those instances of CS such that for any composition S of cardinality greater than n, S exhibits a degree of emergence 0. Each composition of cardinality less than or equal to n may exhibit some varied degree of emergence d, d > 0.*

We will show that the standard Greedy approach to solving *n*-CS is only desirable when *n = 2*, and that for *n > 2*, the Greedy approach is futile. We choose not to portray *n*-CS as a broadly tractable variation of CS, but to expose our line of thinking about variations of CS that do or do not lend themselves to the standard Greedy approach.

For *n*-CS we prepend the Greedy algorithm with the construction of an assistive data structure we refer to as a *composition table*.

Definition 6 *A composition table for an instance of n-CS is a list of pairs, one pair for each composition S of cardinality less than or equal to n. A pair's members are S and σ(S).*

The time required to create a composition table is on the order O(*m* choose *n*) where *m* is the total number of components; therefore, this time is of exponential order. When *n = 2* the time is on the order O(m choose 2), or $O(m^2)$, a polynomial-time complexity class. Therefore, *2*-CS is tractable for creating and using a composition table for approximating its solution.

For an instance of *n*-CS, let $T = \{t_1, t_2, \ldots, t_z\}$ be the set of elements in the composition table and let $O = \{o_1, o_2, \ldots, o_n\}$ be the set of objectives associated with the instance. Algorithm 3 illustrates Greedy on a generic composition table, Table 2, of an instance of *n*-CS.

Table 2: Generic Composition Table

$t_1 = \{ S_1, \sigma(S_1) \}$	$t_2 = \{ S_2, \sigma(S_2) \}$	…	$t_z = \{ S_z, \sigma(S_z) \}$

Algorithm 3
Input: (T, O)
Step 1: $X := NULL;$
$Z := O;$
$i := 1;$
Step 2: *while* $Z \neq NULL$
do begin choose t in T such that $|\{\sigma(X \cup t.S)\} \cap Z|$ *is maximal;*
$X := X \cup \{t.S\};$
$Z := Z - t.\sigma(S);$
end;
Output: $X;$

Theorem 4 *Algorithm 3 is Har(max{|t.σ(S)| | t ∈ T)-approximate for n-CS.*

Proof of Theorem 4 By definition, emergent behavior only occurs amongst compositions of size *n* or less. Additional emergence for compositions greater than size *n* is not possible. All emergence for compositions of size *n* or smaller are stored in the table. Combining elements of the table will produce compositions of size larger than *n* and therefore no additional emergent behavior could possibly take place. The table is an instance of MSC with σ(*S*) of each member of the table being an *S* in MSC. Thus, the approximation ratio for Greedy on MSC applies exactly to Algorithm 3 on *n*-CS, and therefore Algorithm 3 is *Har*(max{|*t*.σ(*S*)| | *t* ∈ *T*)-approximate for *n*-CS.
[End of Proof of Theorem 4]

Note that *2*-CS is the same as a variation of *u*-UCS, where *u* varies for every PS in an instance over the range [*0*, |*O*|] and so we have shown how to approximate such a variation.

5 ON THE INAPPROXIMABILITY OF CS

Proving the inapproximability of an NP-complete problem means proving a lower bound on how well the problem can be approximated. We present here a conjecture that CS, as defined in (Petty, Weisel, and Mielke 2003) is inapproximable to a practical factor because the nature of emergence in the problem is unpredictable. We remind the reader that the CS defined in (Petty, Weisel, and Mielke 2003) includes anti-emergence and makes no assumptions about the nature of emergence.

Three methods of proving the inapproximability of an NP-complete problem *U* are (Hromkovic 2001):

1. Reduce a known NP-hard decision problem to the approximation problem – the problem of finding a feasible solution within a fixed approximation ratio *r* – associated with *U*.
2. Reduce a problem known to be inapproximable to *U* such that the reduction preserves the approximation ratio.
3. Apply the PCP-(Probability Checkable Proofs) Theorem to show that some decision problem in NP is reducible to the approximation problem associated with *U*.

We believe that CS is practically inapproximable because it seems that for an approximation algorithm to be useful on an instance of CS it must have prior knowledge of the nature of emergence in the instance (i.e. it must know the degree of emergence and anti-emergence for all the possible compositions in the problem). Obtaining this prior knowledge induces too much cost in running the algorithm, forcing it into a higher complexity class than that of any polynomial-time complexity class. We believe that there exists an NP-hard decision problem that is reducible to the approximation problem associated with CS, thus implementing method 1 in showing CS is inapproximable. Furthermore, the result of that proof will reveal an impractical lower bound on any approximation ratio. We leave the establishment of this belief to future work.

6 CONCLUSION

Even though many researchers have proposed constructions for composability, our research elucidates the inherent computational complexity in all of the proposed constructions. If our conjecture on inapproximability is correct, the inherent complexity of composability will prohibit the future realization of systems of composability as typically envisioned. It is necessary to understand from whence the complexity derives and how it can be constrained. Our results provide a first step towards demonstrating a formal process for forcing CS into approximable variations. This paper has shown two such variations: one in which components exhibit equal and uniform emergence (*u*-UCS) and the other in which only pair-wise compositions exhibit emergence (*2*-CS). These variations provide the first framework for a reduction of complexity to which engineers of a sound and computationally practical system of composability would adhere.

7 FUTURE WORK

We envision two paths to take regarding future work on the complexity problem CS:

1. Design, implement, and test heuristic algorithms for solving CS.
2. Prove the conjecture of the inapproximability of CS.
3. Design, implement, and test heuristic algorithms for solving CS.
4. Prove the conjecture of the inapproximability of CS.

Taking path 1 would lead to extended methods for approximating CS. However, designing good heuristic algorithms requires a robust testbed of CS instances, which does not yet exist. Taking path 2 would lead to important theoretical results concerning the impossibility of composability as it is commonly envisioned. Proving the conjecture would force composability experts to rethink CS and realize that without serious restrictions on the rules of component engineering, such as forcing a set of components to exhibit uniform emergence, composability is unachievable.

REFERENCES

Cormen, T. H., C. E. Leiserson, R. L. Rivest, and S. Clifford. 2001. *Introduction to Algorithms* (2nd ed.). Massachusetts: MIT Press.

Dahmann, J. S., J. O. Calvin, and R. M. Weatherley. 1999. A reusable architecture for simulations. *Communications of the ACM* 42:79-84.

Hromkovic, J. 2001. *Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Berlin: Springer.

No author given. 2002. *DMSO MSRR Homepage.* [online] Available online via: <http://www.msrr.dmso.mil> [accessed July 12, 2004].

Page, E. H. and J. M. Opper. 1999. Observations on the complexity of composable simulation. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 553 – 560. New York: ACM Press. Pheonix, Arizona.

Petty, M. D., E. W. Weisel, and R. R. Mielke. 2003. Computational complexity of selecting components for composition. *Proceedings of the Fall 2003 Simulation Interoperability Workshop*, Orlando FL, September 14-19.

Slavík, P. 1996. A tight analysis of the greedy algorithm for set cover. *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, Philadelphia, PA.

Wang, N., D. C. Schmidt, and C. O'Ryan. 2000. "An Overview of the CORBA Component Model." *Component-Based Software Engineering*, Addison-Wesley.

AUTHOR BIOGRAPHIES

MICHAEL ROY FOX is an Undergraduate Student at the Computer Science Department at the University of Virginia. His research interests include composability, simulation coercion, and multi-resolution modeling. His email address is <mrf4u@cs.virginia.edu>.

DAVID C. BROGAN earned his Ph.D. from Georgia Tech and is currently an Assistant Professor of Computer Science at the University of Virginia. For more than a decade he has studied simulation, control, and computer graphics for the purpose of creating immersive environments, training simulators, and engineering tools. His research interests extend to artificial intelligence, optimization, and physical simulation. His email address is <dbrogan@cs.virginia.edu>.

PAUL F. REYNOLDS, JR. is a Professor of Computer Science at the University of Virginia. He has conducted research in Modeling and Simulation for over 25 years, and has published on a variety of M&S topics, including parallel and distributed simulation, multi-resolution modeling and coercible simulations. He has advised industrial and government agencies on matters relating to modeling and simulation. He is a plank holder in the DoD High Level Architecture. His email address is <reynolds@cs.virginia.edu>.