Modularizing Global Variable In Climate Simulation Software

Tapajit Dey, Yuxing Ma, and Audris Mockus
Department of Electrical Engineering and Computer Science
The University of Tennessee, Knoxville

Abstract

In simulation codes, such as climate models, variables represent a large number of characteristics of earth surface and atmosphere for a single multi-dimensional cell and are distributed over a multitude of cores in the supercomputers where these simulations run. The hundreds of variables allow different parts of simulation representing specific sub-models, for example the photosynthesis, to interact with other sub-models of the simulation.

The scientists of each domain write the simulation code for the sub-model representing their sub-domain. To integrate their code into the entire simulation, they need to deal with hundreds of unfamiliar variables of which only a small subset is relevant to their work. Designing such variables in a modular fashion, so that the scientists could interact only with the variables relevant to their sub-model is likely to increase the productivity of the scientists and to increase accuracy of the simulation codes.

A natural way to group the variables into modules is by using a language feature that group them together, such as, struct construct in C language or a class in C++ language. Each scientist would then need to familiarize themselves with only a small subset of modules that contain variables used in their simulations. For example, Community Earth System Model (CESM) code v1.06 has 51 such modules (structures) that contain 1479 variables. The methods proposed below can be used to assess the modularity of the existing set of structures and to generate alternative modularizations that improve upon it.

In a nutshell, the approach minimizes the number of variables exposed to other domains by the modules used in each domain.

Background/Research to Date

To describe the approach in more detail, we need to define several concepts precisely.

- v_i: is a variable used in one or more domains of the simulation.
- A module m_j is a subset of variables. Modules partition the set of variables.
- A domain d_k is a sub-model representing a specific biological, physical, or other type of process. Each d_k reads from or writes to one or more variables v_i. Unlike for modules, the same variable may belong to multiple domains.
- m_j is *necessary for domain* $d_k \iff \exists v_i : v_i \in d_k \wedge v_i \in m_j$.
- v_i is visible from d_k if it belongs to any of the modules m_j necessary for d_k.

Each partition of v_is induces a set of modules. We can measure the quality of the partition by counting the variables visible from all domains:

$$\sum_k | \cup_{m_j \; necessary for \; d_k} | \tag{1}$$

The lower bound for this criteria is simply the number of variables. For example, if the unions of modules necessary for each domain form a partition (do not overlap for any two domains). If, however, any domain uses at least one variable from another domain, this bound may not be achievable. A trivial way to achieve the minimum for the criteria in Equation 1 over all possible partitions given a set of domains is to create a separate module for each variable.

The criteria then becomes: $\sum_k |\cup_{v_i \in d_k}|$. This, however, creates far too many modules defeating the original goal of modularity. It is, therefore, important to note that the criteria should be compared for modularizations that bound the total number of modules or consider Pareto-optimal (non-dominated) solutions defined by minimizing the number of modules and the value of the criteria.

We applied the approach to CESM code (v1.06) and discovered 216 domains (functions) with the optimal (trivial) partition with 1479 modules (each corresponding to one variable) yielding the score (from Equation 1) of 3703. The existing partition of CESM code into the 51 modules (C structures) has the score of 35703. This means that on average, a domain scientist sees ten times more variables than they need to. A partition that contains all variables in a single module, on the other hand, would have the score of 216×1479 or nine times more than the existing partition and 90 times more than the optimal partition.

Proposed Direction of work

The simplest way to resolve the contention between having too many modules and too many variables visible to a domain scientist is to add a penalty term to Equation 1. The penalty would increase with the number of modules used to partition the variables. Initially we will optimize the criteria for each fixed number of modules to find non-dominated solutions and then decide which of them is the most practical one. We plan to use several approaches, for example, a modification of the algorithm used to create independently changeable parts of code (modules) in a very large telecommunications system [1]. The algorithm iteratively redistributes these variables by removing or inserting a variable into module or exchanging a variable between two modules. If the move improves the score it is taken if not it is accepted with a probability p that goes down with iterations (annealing). Using this method, the score decreases from 35703 to 29982 after 10 minutes on a single-core cpu. The probability p and the annealing rate can be optimized to achieve faster convergence. The results, while asymptomatically optimal, may yield a different solution with each finite run. The search may be further improved with an initial partition that has no overlaps (where each domain needs only variables from a single partition.) In CESM code we found six non overlapping partitions. Only five exiting modules cross this partition boundary.

Connections to Math, Comp Sci & and Climate Science

The proposed work targets the domain of complex simulation codes, such as ones used in climate science, where multiple phenomena are being modeled with very large numbers of variables used to tie these sub-models together. We plan to apply the approach to several climate simulation codes investigate how data modularity changes over time and could be improved. We plane to interview domain scientist to understand better how data modularity could address their needs and quantify the impact of improved modularity.

Potential Impact on the field

Existing climate models and other-large-scale simulations require contributions of scientist from many different domains, who are familiar with the sub-models relevant to their domain. Existing simulation codes, however, expose them to thousands of variables representing unrelated phenomena. A suitable modularization these data structures may significantly increase the productivity of scientist and lead to more accurate simulations.

References

[1] Audris Mockus and David M. Weiss. Globalization by Chunking: a Quantitative Approach. IEEE Software, 18(2):30-37, March 2001.