Some Organising Principles for Coupling in Multiphysics and Multiscale Models

J. Walter Larson*†‡§

August 31, 2006

Abstract

Computational science faces new challenges posed by multiphysics and multiscale, or more generally put, *coupled* models. These systems are composites formed from separate subsystem models that interact via data exchanges. These data dependencies pose a *coupling problem*, and on distributed-memory computers, a *parallel coupling problem*. This paper presents a definition of terms and a set of organising principles for the coupling and parallel coupling problems. It is meant as a first step towards creating a theory of coupled models. These principles are then employed in a case study of a coupled climate model and offer remarkable insight into its structure.

Contents

1 Introduction **2**

2 Definition of Terms and Problem Statement **3**

3 The Organising Principles **4**

4 Case Study: Coupled Climate Modelling **8**

*ANU Supercomputer Facility, The Australian National University, Canberra, Australia

†Australian Partnership for Advanced Computing (APAC)

‡Mathematics & Computer Science Division, Argonne National Laboratory, Argonne, Illinois, USA

§Computation Institute, University of Chicago, Chicago, IL, USA

5 Conclusions **10**

1 Introduction

Computational science is becoming more ambitious by moving beyond the traditional approach of simulating individual isolated subsystems towards integrated systems having numerous mutually interacting components. Two distinct types of these composite models are emerging: *multiphysics* models, which violate the frequent modelling assumption that the system under study does not interact with the outside world; and *multiscale* models, which violate an often-imposed notion that phenomena prevalent on disparate spatiotemporal scales do not interact. The main driver for this change of approach has been the advent of high-performance computing, and in particular, message-passing parallel computing (a.k.a. distributed-memory parallelism) on commodity microprocessor-based clusters.

A classic example of a multiphysics model is a climate system model, comprising an atmospheric general circulation model (GCM), an ocean GCM, a fully dynamic sea-ice model, and a land-surface model[1]. Other multiphysics modelling problems can be found in the fields of controlled thermonuclear fusion, space weather, reactive flow, modelling of rocket engines, fluid-structure interaction, materials science, and groundwater hydrology.

Numerical weather prediction provides a leading example of a multiscale application in forecast models that allow multiple, nested, and interacting computational domains, such as the Weather Research and Forecasting (WRF) Model[5]. Examples of multiscale systems abound in science and engineering in the fields of plasma physics, climate and weather, biology, hydrology, and materials science.

Multiscale and multiphysics models are *coupled models*, a term adopted from climate modelling that describes well the importance of model-to-model interaction in these systems. Some software technology has been developed to support coupling, with many application-specific ad hoc solutions, or in some cases slightly more general domain-specific packages. More generic coupling infrastructure packages exist[4, 3]. Coupling and parallel coupling form a computational science problem in need of a precise definition and some theoretical foundations. This paper is an early attempt to construct a vocabulary for describing coupling and parallel coupling, and to state some organising principles. Taken together they are not yet a rigorous theoretical framework, but rather a set of heuristic notions whose explicatory power will be demonstrated in Section 4.

2 Definition of Terms and Problem Statement

A *coupled model* $\mathcal{M}$ consists of N constituent[1] models—or simply constituents—that collectively model a complex system through their evolution and mutual interactions.

A *constituent* $\mathcal{C}_i$ is characterised by a *model* M_i that solves its equations of evolution on its *domain*[2] Γ_i to calculate *state* U_i. This state is computed using the current model state and a set of *input variables* V_i. *Output variables* W_i are computed from U_i. The sets V_i and W_i comprise the connections of $\mathcal{C}_i$ to the outside world, and are defined on the *boundary domain* $\partial\Gamma_i$ (or subset thereof). Thus, a constituent $\mathcal{C}_i$ is $\mathcal{C}_i \equiv \{M_i, U_i, V_i, W_i, \Gamma_i, \partial\Gamma_i\}$.

Two components $\mathcal{C}_i$ and $\mathcal{C}_j$ are *coupled* if and only if (1) $\Gamma_i \cap \Gamma_j \neq \emptyset$ and (2) (a) $W_j \cap V_i \neq \emptyset$ and/or $V_j \cap W_i \neq \emptyset$ or (b) the inputs V_i (V_j) can be computed from the outputs W_j (W_i). That is, two models are coupled if their domains intersect and some of the outputs of one model serve as some of the inputs to the other. Coupling between $\mathcal{C}_i$ and $\mathcal{C}_j$ occurs on the *overlap domain* $\Omega_{ij} = \Gamma_i \cap \Gamma_j$ (Figure 1). Couplings are transformations $T_{ij} : (W_{ji}, \Omega_{ij}) \rightarrow (V_{ij}, \Omega_{ij})$ and $T_{ji} : (W_{ij}, \Omega_{ij}) \rightarrow (V_{ji}, \Omega_{ij})$ that deliver inputs to $\mathcal{C}_i$ and $\mathcal{C}_j$, respectively.

The above definitions collectively specify a coupled model $\mathcal{M}$ as $\mathcal{M} \equiv \{\mathcal{C}_1, \ldots, \mathcal{C}_1, \mathcal{T}\}$, where $\mathcal{T} \equiv \{T_{ij}, i = 1, \ldots, N, j = 1, \ldots, N, i \neq j\}$ is the set of all the inter-constituent coupling transformations.

Domain overlap can range in severity from the simplest case of a lower-dimensional interface (Figure 1(a)) to partial colocation (Figure 1(b)) to complete colocation ($\Gamma_i = \Gamma_j$). In principle multiple domains can intersect, forming *higher-order overlap domains*. In Figure 1(c), three domains Γ_i, Γ_j, and Γ_k share such a domain $\Omega_{ijk} = \Omega_{ij} \cap \Omega_{jk}$. In this configuration, *merging* of two constituents' ouptuts for subsequent input is required on Ω_{ijk} if (1) $W_i \cap W_j \cap V_k \neq \emptyset$, or (2) $W_j \cap W_k \cap V_i \neq \emptyset$, or (3) $W_k \cap W_i \cap V_j \neq \emptyset$.

The overlap domains together form the boundary domain of a constituent. That is, $\partial\Gamma_i = \cup_{i \neq j} \Omega_{ij}$. The *interior domain* $\hat{\Gamma}_i$ of $\mathcal{C}_i$ is the part of the domain that is not in direct contact with the other constituents in $\mathcal{M}$, and is the set complement of Γ_i relative to its boundary $\partial\Gamma_i$. That is, $\hat{\Gamma}_i = \Gamma_i \backslash \partial\Gamma_i$.

[1] Many authors use the term *component* to label the individual parts of a coupled model. Component-based software engineering is now emerging as a key software technology for these systems, and a software 'component' is not necessarily the same as a model 'component.' For this reason, I will eschew the use of the term component in favour of constituent.

[2] In principle a model might have multiple domains, but for illustrative purposes, it is assumed here that each model has only one domain. The ideas presented in this paper can be extended to the case of multiple domains.

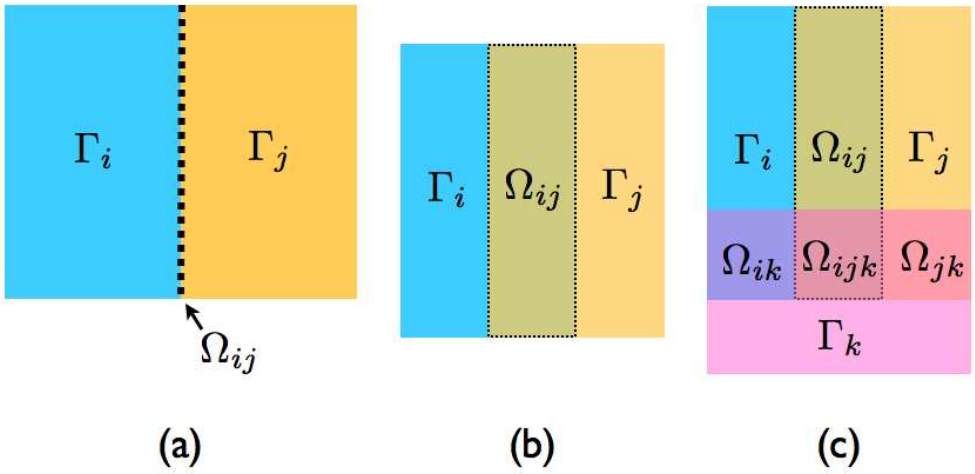

Figure 1: Overlap domain configurations in two dimensions: (a) a one-dimensional interface, (b) partial colocation, and (c) three intersecting domains with multiple overlap domains.

In building a coupled model on a traditional uniprocessor (von Neuman) computer architecture, one must surmount the following obstacle:

Problem 1 (Coupling Problem) *Given N models executing in mutual interaction, create a working coupled model.*

3 The Organising Principles

Aspects of the CP and PCP The coupling problem (CP) and parallel coupling problem (PCP) share an immediate organising principle in terms of process decomposition.

Organising Principle 1 *The CP and PCP each may be decomposed into challenges of (1) coupled model architecture, (2) data processing in aid of coupling, and (3) software environment.*

This paper focuses on the architectural and algorithmic aspects of the CP and PCP, which correspond to the first two aspects identified in this organising principle. Software environment encompasses the bridging of programming language barriers and software build strategies to arrive at a working executable from source code, and as such are beyond the scope of this discussion.

Connectivity Graph theory[6] is an often-used tool for describing systems with interdependent processes, leading us to a fundamental organising principle for the CP.

Organising Principle 2 *A coupled model $\mathcal{M}$ can be represented as a directed graph G, and this digraph is connected.*

In this graph-theoretic picture of coupled models, the constituents and their data dependencies are represented as nodes and arcs, respectively. A coupled model's associated digraph G is connected because were it not, G would then consist of two or more separate graphs, implying $\mathcal{M}$ could be separated into two or more independent coupled systems. Figures 2(a) and 2(b) depict directed-graph representations of coupled systems having four and five constituents, respectively. A dependency of node A on output from node B is expressed by an arc pointing from node B to node A. Each node's associated model computes its state from its time history, combined with its inputs. This self-dependence could in principle be signified by one or more *loops* on each node. Here the convention will be not to include loops.

The *in- (out-) valency* of a node is equal to the number of incoming (outgoing) data connections from (to) the corresponding constituent in the coupled system. In a digraph, there are five possible distinct connectivity relationships between any two nodes, and each of these corresponds to a different data dependency relationship between $\mathcal{C}_i$ and $\mathcal{C}_j$: $\mathcal{C}_i$ receives (delivers) direct input (output) from (to) $\mathcal{C}_j$ (*direct coupling*); $\mathcal{C}_i$ does not receive (deliver) data directly from (to) $\mathcal{C}_j$, but instead via a *path* through a series of one or more intermediate constituents (*indirect coupling*); and $\mathcal{C}_i$ and $\mathcal{C}_j$ have no path connecting them and are thus *decoupled*. If a node has only incoming (outgoing) arcs, it is called a *sink* (*source*), and its associated constituent *can be run off-line*. If $\mathcal{C}_i$ is associated with a source it can be run off-line and its time history can be fed to the rest of the coupled system at a later time. If $\mathcal{C}_i$ is associated with a sink, the rest of the coupled system can be run first, and its time history subsequently can be fed to $\mathcal{C}_i$.

The connectivity of a parallel coupled model $\mathcal{M}$ is the list of direct inter-component interactions, and in a graph-theoretical context is expressible as the *adjacency matrix* $\mathbf{A}$ of its associated digraph G.

Scheduling of Coupling Events Coupled models evolve by solving their constituents' equations, a process in which inter-constituent data exchanges play a key role.

Organising Principle 3 *Coupling events between any two constituents can occur either following a* **schedule** *with the coupling event times known a*

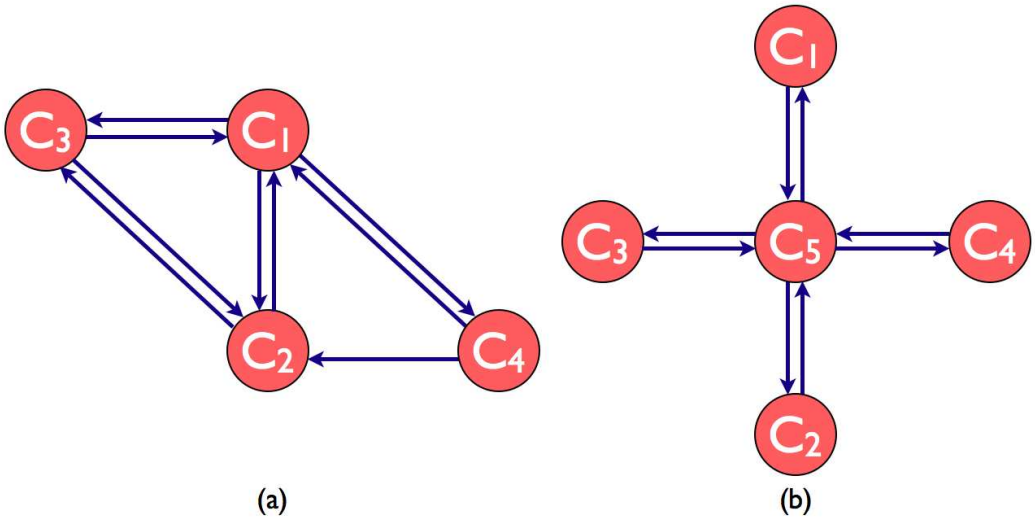

Figure 2: Directed graphs for coupled systems with (a) four constitutents, and (b) five constituents.

priori, or in a potentially nonperiodic and unpredictable fashion **triggered by some threshhold**.

If the exchange periods between all the constituents are mutually commensurate and not displaced in time by offsets incommensurate with the exchange periods, one can define a repeatable *coupling cycle* and, within this cycle, a *coupling frequency* for each inter-constituent exchange. Coupling frequency is determined by the interconstituent coupling sensitivities, and the timescales over which the constituents evolve significantly. In practice, these frequencies are often chosen based on intuition and experimentation.

Coupling Strength Coupling strength between two constituents $\mathcal{C}_i$ and $\mathcal{C}_j$ can be characterised by the *sensitivity* of their states U_i and U_j to the couplings, the degree of colocation between their domains, and the computational overhead due to coupling.

The most compelling measure of coupling strength between constituents $\mathcal{C}_i$ and $\mathcal{C}_j$ is the impact of W_i on U_j and W_j on U_i. Of particular interest is the impact of the couplings on U_i and U_j within $\hat{\Gamma}_i$ and $\hat{\Gamma}_j$, respectively.

Organising Principle 4 *For a constituent $\mathcal{C}_i$, overall coupling sensitivities $\boldsymbol{\sigma}_i$ are defined by the Jacobian of its state U_i with respect to its inputs V_i, $\mathcal{J}_i = \partial(U_i;\hat{\Gamma}_i)/\partial(V_i;\partial\Gamma_i)$, and $\boldsymbol{\sigma}_i = \mathcal{J}_i$. The coupling sensitivities $\boldsymbol{\sigma}_{ij}$ of $\mathcal{C}_i$, to input from $\mathcal{C}_j$ are $\boldsymbol{\sigma}_{ij} = \mathcal{J}_{ij} = \partial(U_i;\hat{\Gamma}_i)/\partial(V_{ij};\Omega_{ij})$.*

Two constituents $\mathcal{C}_i$ and $\mathcal{C}_j$ may have *direct* dependencies between their respective states U_i and U_j. *Diagnostic coupling* occurs on Ω_{ij} when (1) $W_i \cap U_i = \emptyset$ and $V_j \cap U_j = \emptyset$ and (2) $W_j \cap U_j = \emptyset$ and $V_i \cap U_i = \emptyset$. *Prognostic coupling* occurs on Ω_{ij} when (1) $W_i \cap U_i \neq \emptyset$ and $V_j \cap U_j \neq \emptyset$ and/or (2) $W_j \cap U_j \neq \emptyset$ and $V_i \cap U_i \neq \emptyset$. Examples of diagnostic coupling are exchanges of either boundary conditions or interfacial fluxes such as in a coupled climate model. Prognostic coupling is indicative of stronger coupling such as self-consistent computation of electromagnetic fields for magnetosphere-ionosphere coupling.

Organising Principle 5 *To the coupled model builder, diagnostic coupling is preferable because it is easier to implement. Prognostic coupling can impose a requirement for repeated iterative execution of constituents to achieve self-consistent state solutions.*

Thus far, we have treated constituent domains generally as *sets*. In practice, Γ_i is a discretised finite subset of $\Re^D$, and thus both Γ_i and $\partial\Gamma_i$ will be countable finite sets. Let $S(\Gamma)$ be the number of elements in a domain Γ. This definition leads to an effective surface-to-volume ratio ρ_i for $\mathcal{C}_i$.

Organising Principle 6 *The domain surface-to-volume ratio ρ_i for a constituent $\mathcal{C}_i$ is $\rho_i = S(\partial\Gamma_i)/S(\Gamma_i)$.*

Each constituent can have its own domain discretisation. Therefore, $\mathcal{C}_i$ and $\mathcal{C}_j$ can have differing discretisations of Ω_{ij}, with $S_i(\Omega_{ij})$ and $S_j(\Omega_{ij})$ elements as seen by $\mathcal{C}_i$ and $\mathcal{C}_j$, respectively.

Organising Principle 7 *The degree Δ_{ij} of colocation of two domains Γ_i and Γ_j is $\Delta_{ij} = [S_i(\Omega_{ij}) + S_j(\Omega_{ij})]/[S_i(\Gamma_i) + S_j(\Gamma_j)]$.*

Coupling overhead can be assessed by analysing the system's *load matrix* $\mathbf{L}$. The off-diagonal elements L_{ij} are the cost of performing the transformations $T_{ij} : (W_j, \Omega_{ij}) \rightarrow (V_i, \Omega_{ij})$, and the diagonal elements L_{ii} are the cost of evolving the constituents $\mathcal{C}_i$ in decoupled mode. These costs are typically defined in terms of computer resources (e.g., CPU time).

The elements of $\mathbf{L}$ can be used to compute a number of execution cost metrics. The system-wide decoupled simulation cost is $K_D = \mathrm{Tr}(\mathbf{L})$. The coupling cost to $\mathcal{C}_i$ is $K_{Ci} = \sum_{i \neq j}(L_{ij} + L_{ji})/2$. The total coupling cost to $\mathcal{M}$ is $K_C = \sum_{i=1}^{N} K_{Ci}$. The total coupled simulation cost is then $K_T = K_C + K_D$.

Organising Principle 8 *Coupling tightness can be quantified in terms of the ratio of coupling cost to total simulation cost.*

The *coupling overhead* ω_i imposed on $\mathcal{C}_i$ is $\omega_i = K_{Ci}/(L_{ii} + K_{Ci})$. The *coupling tightness* τ_{ij} between $\mathcal{C}_i$ and $\mathcal{C}_j$ is $\tau_{ij} = (L_{ij} + L_{ji})/(L_{ii} + L_{jj} + L_{ij} + L_{ji})$. The *total coupling overhead* τ is $\tau = K_C/K_T$.

Effects of Distributed Memory Parallelism The chief impetus for model coupling is the computational capacity created by the message-passing parallel programming model. On such platforms, the lack of a global address space poses a different coupling problem:

Problem 2 (Parallel Coupling Problem) *Given N models that employ distributed-memory parallelism and executing in mutual interaction, create a working and scalable parallel coupled model.*

Organising Principle 9 *The parallel coupling problem (PCP) is a superset of the challenges posed by the coupling problem (CP). The definitions and organising principles stated thus far apply equally well to the CP and PCP.*

Distributed memory increases coupled model architectural complexity by introducing *concurrency*. In the CP, the constituents and their couplings can execute only in turn as an event loop (*serial composition*). In the PCP, concurrency allows another strategy called *parallel composition*[2] in which the global processor pool is partitioned into *cohorts*, one for each constituent. This allows the constituents to execute simultaneously. Parallel composition has the advantage in minimising processor idle time by choosing cohort size based on its constituent's parallel scaling behaviour. The disadvantage of parallel composition is that coupling becomes sensitive to synchronisation between independently running parallel models, making performance tuning notoriously difficult. Serial and parallel composition strategies can be combined (*hybrid composition*).

Data processing operations for the PCP are *parallel operations*. This requires the description of *distributed data* (i.e., V_i and W_i and their resident domains Γ_i and $\partial\Gamma_i$). The coupling transformations T_{ij} are now message-passing parallel operations, and in addition to computation they will likely be required to perform parallel data *transfer* and/or *redistribution*.

The parallel data description, transfer, and transformation operations described above are amenable to automation, and a prime example of such technology is the Model Coupling Toolkit (MCT)[4, 3]. The architectural decision space created by including concurrency in coupled systems is less well understood, and is a promising topic for further research.

4 Case Study: Coupled Climate Modelling

The Community Climate System Model (CCSM) is a coupled climate model with five constituents $\{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4, \mathcal{C}_5\} = \{$atmosphere, ocean, sea-ice, land, coupler $\}$, forming a hub-and-spokes system (Figure 2(b)). The coupler is an

intermediary through which the other constituents exchange fields, performs the appropriate transformations $\mathcal{T}$, and acts as an overall coordinator for the system's evolution. The coupler exists because (1) some of the model outputs are not precisely the inputs required by the other models (here our second criterion for coupling is active), and (2) it lowers the coupling overhead experienced by the other constituents by consolidating their interactions with the outside world. In the absence of the coupler, the other constituents would have to interact directly in the point-to-point pattern shown in Figure 2(a).

The overlap domains in the model coincide with the earth's surface. All of the models have effectively three-dimensional domains, minimising colocation (e.g., $\rho_1 \approx 0.038$, $\rho_2 \approx 0.025$, and $\Delta_{12} \approx 0.037$). Along coastlines and in regions where sea-ice is present, the interface to the atmosphere will see input of the same fields from multiple entities. Thus higher-order overlap domains exist on which merging is required.

Coupling occurs on a schedule, with a model day as the coupling period. The atmosphere, sea-ice, and land models exchange data each model hour, and diagnostics are time-integrated for coupling with the ocean once per model day. The hourly exchanges between atmosphere, land, and sea-ice are based on the requirement for input radiation fluxes by the atmosphere's radiative transfer package, and because the atmosphere evolves significantly on this timescale. The ocean surface evolves on a slower timescale, and trial and error has arrived at one day for the ocean's coupling frequency.

Computation of the sensitivities $\boldsymbol{\sigma}_i$ and $\boldsymbol{\sigma}_{ij}$ for this model is not practical. Each of the domains has thousands of grid points, and the associated Jacobians are quite large matrices[3]. Furthermore, calculating $\boldsymbol{\sigma}_i$ and $\boldsymbol{\sigma}_{ij}$ is more computationally complex than solving the systems' model equations. Scientists rely instead on visualisaiton-based analysis of model history output (and statistical moments thereof) to assess model sensitivity.

CCSM's couplings are predominantly diagnostic in nature. This obviates the need for iterative execution to arrive at self-consistent solutions, and coupling is thus the one-way delivery of data from one constituent to another via the coupler.

CCSM employs message-passing parallelism, and is implemented using a parallel composition. This allows the combination of codes with differing scaling behaviour (see Figure 4 in reference [4] for details). The chief performance challenge posed by CCSM is the existence of intermittent delays caused by one constituent awaiting data from another, but this is largely solved by shifting these delays from the models to idle time in the coupler. Overall, the coupling overhead in CCSM is low with $\tau < 0.5$.

[3]For example, in CCSM's standard atmospheric configuration, σ_1 has $\mathcal{O}(10^{11})$ elements.

5 Conclusions

A heuristic set of definitions and organising principles for coupled models has been stated. This work is a first step towards a more comprehensive theoretical framework for the CP and PCP. Each of the principles stated here provides a glimpse of a rich vein in need of exploration. This conceptual framework has been applied successfully to describe the architecture of a coupled climate model. Future work will refine these principles and expand them to encompass the complexities of parallel coupling. It is hoped that the resulting theory will guide the development of future coupled systems.

Acknowledgements: I thank Robert Jacob, Michael Tobis, Jace Mogill and Timothy Mattson for their advice and feedback. This work was funded primarily by the US Department of Energy (DOE) through its Scientific Discovery through Advanced Computing Program Argonne National Laboratory is operated for the DOE by UChicago Argonne LLC under contract DE-AC02-06CH11357. The ANU Supercomputer Facility is funded in part by the Australian Department of Education, Science, and Training through the Australian Partnership for Advanced Computing.

References

[1] W. D. Collins, C. M. Bitz, M. L. Blackmon, G. B . Bonan, C. S. Bretherton, J. A. Carton, P. Chang, S. C. Doney, J. J. Hack, T. B. Henderson, J. T. Kiehl, W. G. Large, D. S. McKenna, B. D. Santer, and R. D. Smith. The community climate system model: CCSM3. *Journal of Climate*, 19(11):2122–2143, 2006.

[2] Ian Foster. *Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering.* Addison Wesley, Reading, Massachusetts, 1995.

[3] Robert Jacob, Jay Larson, and Everest Ong. M×n communication and parallel interpolation in ccsm3 using the model coupling tookit. *Int. J. High Perf. Comp. App.*, 19(3):293–308, 2005.

[4] Jay Larson, Robert Jacob, and Everest Ong. The model coupling toolkit: A new fortran90 toolkit for building multi-physics parallel coupled models. *Int. J. High Perf. Comp. App.*, 19(3):277–292, 2005.

[5] WRF Development Team. Weather Research and Forecasting (WRF) Model Web Site. `http://wrf-model.org/`, 2006.

[6] H. N. V. Temperly. *Graph Theory and Applications.* John Wiley and Sons, New York, 1981.