PARALLELIZATION OF A WATERSHED MODEL—PHASE III: COUPLED 1-DIMENSIONAL CHANNEL, 2-DIMENSIONAL OVERLAND, AND 3-DIMENSIONAL SUBSURFACE FLOWS

JING-RU C. CHENG[1], ROBERT M. HUNTER[1], HWAI-PING CHENG[2], DAVID R. RICHARDS[3], AND GOUR-TSYH YEH[4]

[1]Major Shared Resource Center (MSRC), U.S. Army Engineer Research and Development Center (ERDC), Vicksburg, MS 39180-6199, USA
[2]Coastal and Hydraulics Laboratory, U.S. Army ERDC
[3]Information Technology Laboratory, U.S. Army ERDC
[4]Department of Civil and Environmental Engineering, University of Central Florida, Orlando, FL 32816-2450, USA

Abstract. A parallel watershed code, pWASH123D, based on a first-principle, physics-based model (WASH123D), has been developed to simulate large watershed problems on scalable computing systems. A watershed is conceptualized as a coupled system of one-dimensional (1-D) channel network, two-dimensional (2-D) overland regime, and three-dimensional (3-D) subsurface media. A key feature of watershed models is the capability to model flow processes and interactions among different domains. This paper addresses the parallelization of such a complete suite of coupled watershed systems. The performance of a coupled 1-, 2- and 3-D flow example running on the U.S. Army Engineer Research and Development Center Major Shared Resource Center's parallel computers is investigated.

1. INTRODUCTION

The U.S. Army Corps of Engineers plays a critical role in the Nation's watershed management. The watershed hydrology model development is tasked in the U.S. Army Engineer Research and Development Center (ERDC) System-Wide Water Resources Program (SWWRP) to enable the simulation of a wide range of regional watershed systems. Watershed models simulate major hydrological processes on multiple spatial domains over varied temporal scales with interactions among them spanning from uncoupled to strongly coupled. Different numerical approaches for coupled nonlinear hydrologic processes have been proposed to be efficient and affordable. Yeh et al. [YHC+06] presented a first-principle, physics-based watershed model. According to Yeh's review [Yeh02], HSPF (Hydrologic Simulation Program—FORTRAN) and WASH123D are the only models that include complete media systems. The difference between them is that HSPF, which has dominated the watershed simulations for more than two decades, employs the parametric approach, while WASH123D is based on a first-principle, physics-based approach [YHC+06].

Sponsored by different projects such as the Department of Defense (DoD) Common High Performance Computing Software Initiative (CHSSI), the application projects included in the Comprehensive Everglades Restoration Plan (CERP), and the SWWRP project, the parallelization of WASH123D, i.e., the development of pWASH123D, aims to efficiently simulate the complex regional watershed systems. This paper presents the details of parallel algorithms along with the software design in the development. In Section 2, the mathematical formulation in WASH123D is briefly mentioned, especially the coupling approach. Section 3 details the data structure design, parallelization of I/O, coherent data maintenance, a coupling module facilitated by a coupler development, and parallelization of the computational kernel. The underlying parallel algorithms will be described. In Section 4, experimental results are presented to demonstrate the watershed features and the performance of parallel implementation. Section 5 summarizes the results and discusses the future plans.

2. MATHEMATICAL FORMULATION

Yeh et al. [YHC$^+$06] describes the governing equations of 1-D channel flow, 2-D overland flow, and 3-D subsurface flow as well as different numerical approaches to solve such a complex system in great detail. The 1-D channel flow is governed by 1-D Saint-Venant equations, which include one continuity equation and one momentum equation, and are solved by a diffusive wave approach. The 2-D overland flow is computed by solving the depth-averaged diffusive wave equation with the semi-Lagrangian finite element method (FEM). The well-known Richards' equation describing the 3-D subsurface flow is solved with the Galerkin FEM that can be found elsewhere [YCH$^+$03].

The fluxes between surface and subsurface media are computed by imposing continuity of fluxes and state variables (e.g., overland water depth and subsurface pressure head). If the state variables exhibit discontinuity, then a linkage term is used to simulate the fluxes. Considering the interaction between the 2-D overland (or 1-D channel) and 3-D subsurface flows, the pressures in the overland flow (if present) and in the subsurface media must be continuous across the interface. Thus, the interaction must be simulated by imposing continuity of pressures and fluxes as

$$h^u = h^s \quad \text{and } Q^u = Q^s = \mathbf{n} \cdot \mathbf{K} \cdot (\nabla h^s + \nabla z) \ , \tag{1}$$

where h^u is the water depth[L] in the overland (or in the channel) if it is present, h^s is the pressure head[L] in the subsurface, Q^u is the flux [L^3/L^2/t] from the overland (or the channel) to the interface, Q^s is the flux from the interface to the subsurface media [L^3/L^2/t], and $\mathbf{n}$ is an outward unit vector of the ground surface. The use of a linkage term such as $Q^u = Q^s = k(h^u - h^s)$, while convenient, is not appropriate because it introduces a nonphysical parameter, k. The calibration of k to match simulation with field data renders the coupled model *ad hoc* even though the overland (or the channel routing) and subsurface models are each individually physics-based.

Two cases are considered in the interaction of 1-D channel and 2-D overland flows. If the waters are connected, i.e., channel water stage is higher than the top of channel bank, the following continuity equations exist.

$$q^O = q^C \implies S_1 = \mathbf{n} \cdot \mathbf{V}^O h^O \quad \text{and } H^O = H^C \ , \tag{2}$$

where H^O is the water stage in the overland, H^C is the water stage in the channel, q^O is the outward normal flux of the overland flow, q^C is the lateral flow from overland to channel, S_1 is the normal flux from overland to channel, $\mathbf{n}$ is an outward unit vector (from the 2-D overland regime), $\mathbf{V}^O$ is overland flow velocity, and h^O is overland water depth.

```
Foreach 3-D flow time step ($\triangle t_{3DF}$) do
    Foreach 3-D coupling/nonlinear iteration do
        Foreach 2-D flow time step ($\triangle t_{2DF}$) do
            Incorporate infiltration/seepage for 2-D/3-D coupling
            Foreach 2-D coupling/nonlinear iteration do
                Foreach 1-D flow time step ($\triangle t_{1DF}$) do
                    Incorporate infiltration/seppage for 1-D/3-D coupling
                    Incorporate infiltration/seepage for 1-D/2-D coupling
                    Foreach 1-D coupling iteration loop do
                        Solve linearized 1-D flow equation
                    Endfor
                    Incorporate infiltration/seepage for 1-D/2-D coupling
                    Solve linearized 2-D flow equation
                Endfor
            Endfor
        Endfor
        Incorporate infiltration/seepage for 1-D/3-D coupling
        Incorporate infiltration/seepage for 2-D/3-D coupling
        Solve linearized 3-D flow equation
    Endfor
Endfor
```

FIGURE 1. 1-D/2-D/3-D coupling algorithm in pWASH123D

On the other hand, when the waters are separated, i.e., channel water stage is below the top of the channel bank, water may flow from overland to channel only, and the following equations govern the interaction.

$$q^O = q^C = f(h^O) \implies S_1 = \mathbf{n} \cdot \mathbf{V}^O h^O = f(h^O) \tag{3}$$

where f(h^O)is a prescribed function of h^O given by the shape and width of the channel bank. Since it is allowed in WASH123D that the two channel banks corresponding to a channel node may have different elevations, it is then possible that (2) is used for the interaction through one bank and (3) for the other.

Figure 1 depicts the 1-D/2-D/3-D coupling algorithm used in WASH123D [LCEY04]. Ideally, channel flow, overland flow, and subsurface flow should be strongly coupled within each time-step. However, this would introduce unaffordable computational characteristics because small time intervals may be required for solving 1-D channel routing. To make computation affordable, in WASH123D each 3-D flow-time interval may contain more than one 2-D flow-time interval and each 2-D flow-time interval more than one 1-D flow-time interval. The fluxes through the surface-subsurface interface are updated using (1) for 2-D/3-D and for 1-D/3-D in each 3-D coupling/nonlinear iteration. In each 2-D coupling/nonlinear iteration, the fluxes through the channel-overland interface are computed using (2) and (3) for 1-D/2-D coupling.

3. PARALLEL SOFTWARE DESIGN AND PARALLEL ALGORITHMS DEVELOPMENT

Figure 2 shows the overall data structures, the software tools developed in house, and the software toolkit integration. The pWASH123D main program instantiates the data for the entire simulation, which includes three components and one coupler. Since the watershed domain exhibits partially overlapped meshes as shown in Figure 3, the mapping between different meshes can be easily generated by GMS 6.0. Often this piece of information cannot be provided when coupling of independently developed programs is pursued.

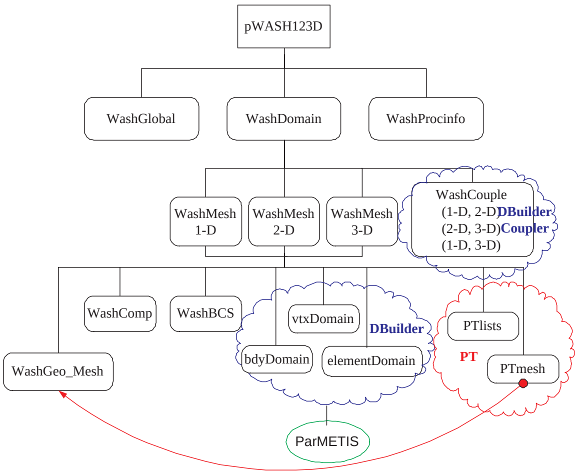

FIGURE 2. Hierarchical data structures designed in pWASH123D

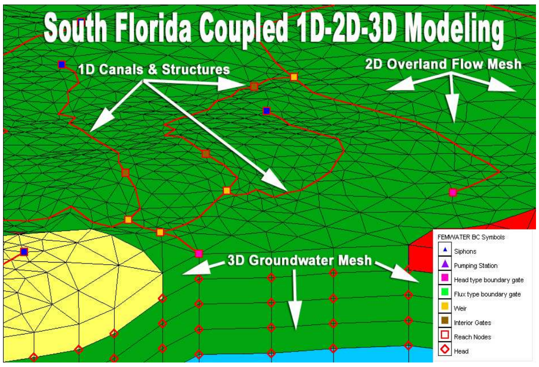

FIGURE 3. Entire simulation domain

3.1. **Hierarchical data Structure in pWASH123D.** The parallel watershed program, pWASH123D, instantiates a `WashGlobal` object describing the common phenomena and setup across the three application components, and a `WashProcinfo` object storing the parallel environmental context. In addition, the object `WashDomain` containing both data and methods embraces the computational domain (Figure 2).

To account for problem domains that may include 1-D river/stream networks, 2-D overland regimes, and 3-D subsurface media, three `WashMesh` objects are created. Each object describes the three subdomains, on which a set of partial differential equations (PDEs) govern the the flow processes, within the entire domain. For 2- and 3-D meshes, the program reads evenly divided numbers of vertices and elements on each processor from a mesh file containing the entire global domain. Each processor constructs its own neighbor list for the parallel graph partitioner, ParMETIS [lbGK], which is actually called in DBuilder [HC05], a parallel data management toolkit developed in-house. Each subdomain is partitioned, based on its favorite partitioning criteria, to processors by DBuilder. Hence, each `WashMesh` object may include `vtxDomain`, `elementDomain`, and `bdyDomain`, which are created and managed by DBuilder, to maintain consistent data structures among processors via ghost vertices/elements on a given mesh. However, every processor reads the entire 1-D mesh from a mesh file and owns them without partitioning.

The `WashCouple` may include the `coupler` for (1-D, 2-D), (1-D, 3-D), and (2-D, 3-D) for the computation of interactions, (1) through (3), between two media. The `coupler` serves as a coupler driver supported by the DBuilder Coupler, which hides all the details of the Message Passing Interface (MPI) scheme for map generating, sending, and receiving between meshes for different components. The merit of this approach is that the partitioning dependency between meshes can be avoided. Moreover, the toolkit can be reused and extended to couple independent application programs.

3.2. **DBuilder Development.** A majority of scientific applications requires a computational mesh, which is a discretization form of the spatial domain. A variety of data can be associated with the mesh, represented by sets of vertices, edges, or elements. When the parallelism is employed, the key is to partition the mesh evenly across processors, maintain consistent data among processors, and implement efficient parallel algorithms to reduce communication overhead. DBuilder was thus developed to provide a simple set of Application Programming Interfaces for users to avoid the work of learning MPI, graph theorem, and parallel algorithms. In fact, the embedded partitioner in DBuilder is ParMETIS.

DBuilder can build a vertex domain, element domain, and boundary domain, each of which is held by an opaque handle named `DB_Subdomain`. For the finite element method, both vertex and element domains are required to maintain a balanced number of vertices among processors. A boundary domain is built to synchronize boundary values. Because the size of boundary vertices/elements is only a small subset of the entire mesh, especially on the 3-D domain, the benefit of using the boundary domain can be a great reduction of communication data size.

3.3. **A Coupler Development for pWASH123D.** There are two types of coupling in pWASH123D. These two types can be characterized as a one-to-one and a one-to-many coupling between domains. First, the one-to-one coupling is utilized where the 2-D mesh is embedded in the larger 3-D simulation. Both meshes are independently partitioned to processors without any constraints. The overlapping region, which is actually the entire 2-D domain, is where information used to solve the systems of equations needs to pass between the two domains. Based on the information of the vertex domain, element domain, or boundary domain that DBuilder constructs and the one-to-one vertex/element mapping that GMS prepares, it is seamless to implement the described coupler functionality in DBuilder. In Figure 4, DBuilder internally represents the 2-D domain as D1 and the 3-D domain as D2, which are set by the ordering of the domains in the `DBuild_Coupler_init` arguments. The function `DBuild_Coupler_update` is called to maintain coherent data between two domains (e.g., D2 and D1 in the example). The demonstration code has the source data on D1 named vecD1 update the data on D2 named `coupler->vec` based on the coupler's element domain specified as the last argument. The third argument gives the data size in bytes for each entry of the vector.

Secondly, one-to-many coupling is used in pWASH123D to map 1-D nodes to multiple 2- and 3-D nodes. The 1-D domain is embedded in the 2-D/3-D domain, but the 1-D domain is not partitioned. Each 2- or 3-D subdomain on a processor has the mapped 1-D domain, but the opposite is not true. GMS provides the information of the 2-D mapped bank nodes associated with each 1-D node. A `DB_Tree` structure regarding the 1-D node as a child and the associated 2-D nodes as parents is built into the (1-D, 2-D) coupler. By providing the 1-D domain as the child domain and 3-D domain as the parent domain, another `DB_Tree` can also be created for the (1-D, 3-D) coupler. Callback functions can be specified by any applications to update vectors from parents to children, and vice versa.

```
/*** Initialization and creation of coupler’s vtxDomain and elementDomain ***/
ierr = DBuild_Coupler_Init(&mesh2->vtxDomain,&mesh3->vtxDomain,vtxMapping,
                 mesh2->vtxDomain.numberLocalElements,&coupler->vtx_coupler);
ierr = DBuild_Coupler_Init(&mesh2->elementDomain, &mesh3->elementDomain,elmMapping,
                 mesh2->elementDomain.numberLocalElements,&coupler->elm_coupler);

/*** get the size of coupler’s element domain ***/
ierr = DBuild_Get_coupler_size(&couplerSize,&coupler->elm_coupler,DB_D2TOD1);

/*** update D1’s (e.g. 2-D) vector to D2’s (e.g. 3-D) vector on the element domain
 ***/
ierr = DBuild_Coupler_update(coupler->vec,vecD1,bytesPerEntry,
                         DB_D1TOD2,&coupler->elm_coupler);
```

Figure 4. Code using DBuilder functions to build a coupler

4. EXPERIMENTAL RESULTS

Figure 5 depicts the example domain, which includes five canal reaches discretized to 590 vertices and 581 elements, a 2-D overland domain with 50,346 vertices and 99,875 elements, and a 3-D subsurface domain discretized to 402,768 vertices and 699,125 elements. Figure 6 details that the 1-D domain also contains four junctions (JT-1 through JT-4), four canal gate structures (G1 through G4), three upstream boundary ends, and two downstream boundary ends. The structures are operated based on the multiple-gate rules. The simulation is made for 24 hours, in which the time-step sizes are 0.1 hours for 3-D, 0.1 minutes for 2-D, and 1 second for 1-D.

Figure 7 plots the wall clock time vs. number of processors for the coupled 1-, 2-, and 3-D flow simulation. A Compaq AlphaServer SC40 machine, configured with 128 nodes connected by a 64-port, single-rail Quadrics high-speed interconnect switch, was used for performance testing. Each node contains four 850-MHz Alpha EV 68 processors and four gigabytes of RAM. From this figure, one can observe that the wall clock time spent on 1-D simulation is nearly constant because each processor, for the n-way run, owns the entire 1-D domain. The majority of runtime is taken up by the 3-D component. The strong scalability is close to linear for 2- and 3-D components (Figure 7) using up to 32 processors. Communication time becomes even more than the ideal wall clock time at the 64-way run (Figure 8). Table 1 indicates that the parallel efficiency is around 84 percent for the 32-way run and less than 30 percent for the 64-way run. Table 1 also lists the times in seconds and in percentage of the total wall clock time associated with 1-, 2-, and 3-D components.

5. CONCLUSION AND FUTURE WORK

A parallel watershed software has been developed to tackle large watershed problems. The software development strategy is an IT-based approach—modular, hierarchical, portable, scalable, and embedded parallel toolkit development and integration. DBuilder has successfully embedded parallelism through the use of MPI to build subdomains for

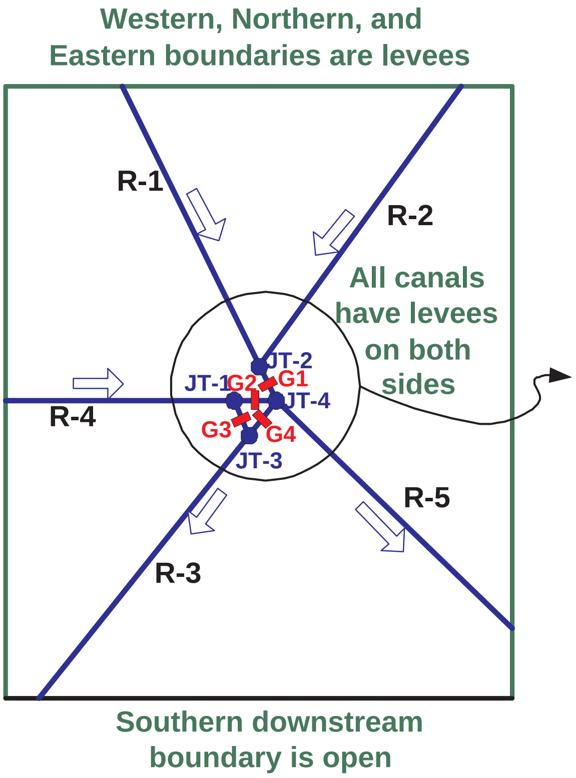

FIGURE 5. Top view of the test domain

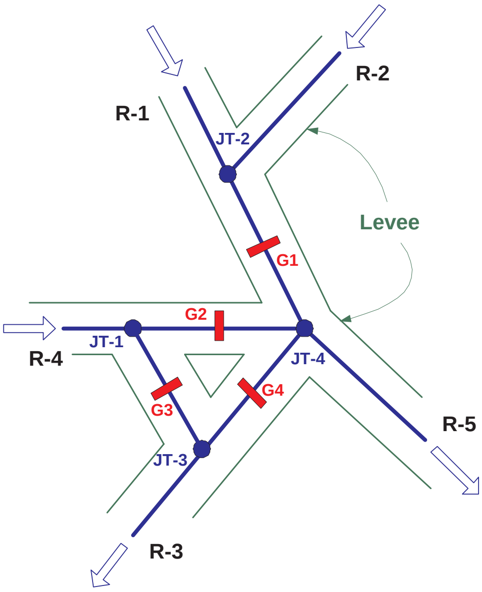

FIGURE 6. Details of the 1-D canal system

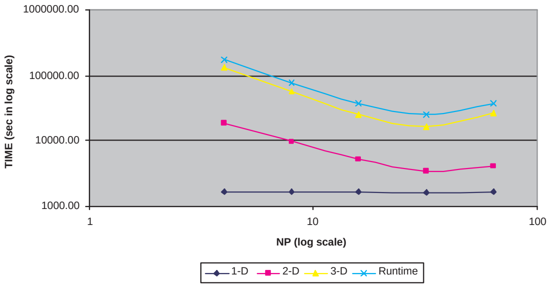

FIGURE 7. Top view of the test domain

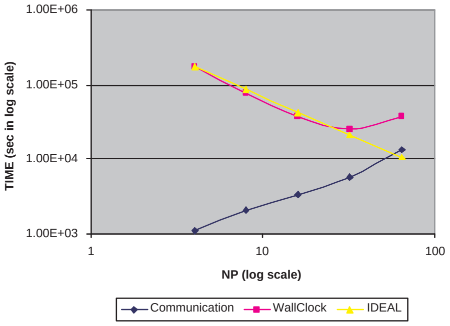

FIGURE 8. Details of the 1-D canal system

each component and two forms of couplers for interactions between components. Experimental results show that the parallel implementation in each component, as well as the coupler development for interactions among components, does improve performance significantly. There is no performance gain at the 64-processor count; this is due to the fact that the problem size is not large enough. Therefore, the time spent in communication becomes greater than the time for computation. Currently, the coupler development relies on GMS or a user-provided one-to-one vertex mapping of the interface between different

Table 1. Scalability on the Compaq SC40: timings (in second)

NP	Runtime	1-D time (%)	2-D time (%)	3-D time (%)	PE
4	173008.14	1669.42 (0.96)	19239.53(11.12)	133523.50 (77.18)	-
8	78460.88	1659.73 (2.12)	9757.95 (12.44)	57726.35 (73.57)	1.1
16	37869.84	1670.90 (4.41)	5240.21 (13.84)	25859.19 (68.28)	1.1
32	25850.92	1623.22 (6.28)	3394.68 (13.13)	16895.63 (65.36)	0.84
64	38090.42	1687.33 (4.43)	4047.45 (10.63)	27185.84 (71.37)	0.28

meshes. Often this piece of information cannot be provided when coupling independently developed models. A parallel algorithm to construct the mapping would be challenging, but a great contribution to model coupling. Furthermore, research to investigate the time-space parallelism on the 1-D domain will be conducted.

ACKNOWLEDGMENTS

This work was supported in part by an allocation of computer time from the Department of Defense High Performance Computing Modernization Program at the U.S. Army Engineer Research and Development Center Major Shared Resource Center, Information Technology Laboratory, Vicksburg, Mississippi. The work was also supported in part by the U.S. Army Engineer Research and Development Center, R&D program of System-Wide Water Resources Project. The authors would like to thank Stephen England at the Philadelphia District, U.S. Army Corps of Engineers for preparing the data set for performance evaluation.

References

[HC05] R. M. Hunter and J.-R. C. Cheng, *DBuilder: A parallel data management toolkit for scientific applications*, Proceedings of the 2005 International Conference on Parallel and Distributed Processing Techniques and Applications (PDATA'05) (Las Vegas, Nevada, USA), CSREA Press, June 27-30, 2005 2005, pp. 825–831.

[lbGK] Research Team led by George Karypis, *ParMETIS parallel graph partitioning*, http://www-users.cs.umn.edu/ karypis/metis/parmetis/.

[LCEY04] H.-C. J. Lin, H.-P. Cheng, E. V. Edris, and G.-T. Yeh, *Modeling surface and subsurface hydrologic interactions in a south Florida watershed near the Biscayne Tay*, Computational Methods in Water Resources XV (The University of North Carolina at Chapel Hill, North Carolina, USA), CMWR CD-ROM (Volume II, II.7.8 Lin_174), June 13-17,2004.

[YCH+03] G.-T. Yeh, H.-P. Cheng, G. Huang, F. Zhang, H.-C. Lin, E. Edris, and D. Richards, *A numerical model of flow, heat transfer, and salinity, sediment, and water quality transport in WAterSHed systems of 1-D stream-river network, 2-D overland regime, and 3-D subsurface media (WASH123D: Version 2.0)*, Tech. Report Draft, Engineer Research and Development Center, U.S. Army Corps of Engineers, MS, 2003.

[Yeh02] Gour-Tsyh Yeh, *A rigorous treatment of interactions between various media in first principle, physics-based flow and transport modeling in watersheds*, EOS Transac., vol. 83, American Geophysical Union, D.C., 2002, p. F436.

[YHC+06] Gour-Tsyh Yeh, Guobiao Huang, Hwai-Ping Cheng, Fan Zhang, Hsin-Chi Lin, Earl Edris, and David Richards, *A first-principle, physics-based watershed model: WASH123D*, Watershed Models (Vijay P. Singh and Donald K. Frevert, eds.), Taylor and Francis Group, LLC, 2006, pp. 211–244.