Plan de Trabajo Final

Ingeniería de Sistemas

Facultad de Ciencias Exactas - UNICEN

Una herramienta para la gridificación de aplicaciones de simulación agropecuaria

5 de noviembre de 2013

Tema: Grid Computing

Alumnos: Rodríguez Alvarez, Juan Maximiliano; Vázquez, Hernán Ceferino.

Director: Dr. Mateos Díaz, Cristian.

Co-Director: Ing. Arroqui, Mauricio.

1. Introducción

La Computación Grid promueve la resolución de problemas complejos mediante la coordinación compartida de recursos en ambientes dinámicos multi-institucionales que comprenden *organizaciones virtuales*. Estos recursos compartidos no sólo involucran archivos, sino también poder de cómputo, software, datos y otros recursos que puedan ser requeridos para la resolución de problemas de la industria, la ciencia y la ingeniería [8]. Particularmente, un Grid computacional (de aquí en más Grid por simplicidad) es un ambiente distribuido que aprovecha múltiples recursos computacionales geográficamente dispersos de forma conjunta para resolver problemas de cómputo intensivo [8]. Así, los recursos por excelencia de un Grid computacional son el poder de cómputo, y secundariamente el almacenamiento y datos. Estos están distribuidos en distintos lugares físicos y son accedidos como si se tratase de una gran supercomputadora "virtual" por parte de los usuarios. De esta manera, una Grid se asemeja a una red eléctrica, que entrega electricidad de forma transparente a partir de centrales eléctricas distribuidas geográficamente. Una propiedad de las redes eléctricas es la facilidad de uso, ya que basta enchufar un electrodoméstico para que éste funcione. Sin embargo, "enchufar" una aplicación a un Grid genera grandes desafíos, y existe una línea de investigación en el área, cuyo objetivo es asistir a los usuarios de aplicaciones a "gridificar" sus aplicaciones a una Grid. Pero, la gran variedad de aplicaciones en cuanto a estructura y funcionalidad hace difícil el desarrollo de un enfoque de gridificación genérico [13].

Entre la variedad de aplicaciones que pretenden beneficiarse del uso de Grid computacionales debido al poder de cómputo requerido se encuentran por excelencia las aplicaciones de simulación en general, y

simulación agropecuaria en particular. Éstas últimas realizan simulaciones mediante el modelado biológico agropecuario de sistemas de la vida real, y presentan una estructura interna y mecanismos de interacción entre sus componentes que hacen que las herramientas de gridificación actuales, tales como BYG [16], EasyFJP [15] o Satin [19], por citar ejemplos representativos, no sean apropiadas [2].

En este contexto, el presente trabajo propone desarrollar una nueva herramienta de gridificación que permita la ejecución paralela y distribuida de aplicaciones de simulación agropecuaria, permitiendo a los usuarios gridificar simulaciones, de forma altamente escalable y eficiente en la utilización de recursos de cómputo. Como atributo de calidad adicional, se pretende que los mecanismos de gridificación desarrollados sean lo menos intrusivos posibles al código biológico secuencial, es decir, permitir a los usuarios gridificar sus simulaciones sin la necesidad de poseer profundos conocimientos sobre Grids.

2. Motivación

En el contexto agropecuario, la investigación disciplinaria se basa en experimentación de campo diseñada para responder a preguntas puntuales. En función de las preguntas que emergen a ese nivel (las disciplinas), se elaboran diseños experimentales concentrados en parcelas y/o períodos de tiempo acotados, generalmente promoviendo la homogeneidad del material experimental (características del suelo, pasturas y animales) y deliberadamente se intenta remover el efecto de otros factores que no son de interés en esos estudios [1]. Este enfoque ha permitido un constante y sólido avance en el conocimiento de los factores que afectan la producción agro-ganadera, pero tiene algunas limitaciones cuando se pretende estimar efectos a nivel sistémico donde los resultados no son causados por un sólo factor [1]. Una alternativa para estudiar dichos efectos a nivel de sistemas es la evaluación en sistemas reales (módulos de producción a escala), pero los mismos resultan costosos en términos monetarios ya que se deben mantener varios años, y el alto costo limita la cantidad de factores a estudiar.

Estas limitaciones pueden ser resueltas mediante simulación, y esta es la razón por la que se ha convertido en una disciplina de investigación de importancia creciente a nivel mundial [17, 20] dada la complejidad sistémica de los modelos. Para dimensionar la complejidad de las interacciones de los componentes del sistema, por ejemplo Romera et al. [18] probaron mediante un modelo diario de una empresa agropecuaria el impacto de diferentes alternativas para reponer las vacas dentro de un sistema de cría de ganados, y encontraron oscilaciones significativas en 50 años de simulación, aún cuando artificialmente se usó un año climático promedio para amenizar el impacto de la incertidumbre climática. Debido al alto número de dimensiones de los sistemas ganaderos, la búsqueda de resultados óptimos también requiere de simulaciones complejas. Aryal et al. [5] optimizaron un sistema ganadero de vacas lecheras. Los autores debieron representar la información inicial en un hipercubo de 8 dimensiones. En definitiva, dado los requerimientos computacionales necesarios, la ejecución en una Grid, potencialmente, ofrece una gran oportunidad para ampliar en gran medida las condiciones de uso y las posibilidades de modelado agropecuario.

La Computación Grid tiene por objetivo conectar, de manera transparente a los desarrolladores, servidores y hasta computadoras personales utilizando redes LAN y WAN para proveer vastos recursos computacionales a las aplicaciones. Esto permite realizar simulaciones más sistémicas. Por ejemplo, la herramienta de simulación APSIM [9] fue gridificada [21] para realizar un cálculo a alta escala y resolución de un modelo de producción del cultivo trigo, carbono del suelo y la dinámica del nitrógeno. Dicho cálculo incluyó 325 escenarios de simulación con un paso de simulación diario durante 122 años. Esta gridificación y posterior ejecución en una Grid permitió realizar un cómputo en sólo 10 días que hubiese tardado 30 años en una máquina. Pero, dicha gridificación, que fue realizada bajo un enfoque API, tiene la desventaja de que el código de la Grid se encuentra entremezclado con la lógica de negocio de la aplicación, requiriendo entonces intervención de usuarios con conocimiento Grid. Además, el mantenimiento, testeo, legibilidad y portabilidad a otras Grids se ven afectadas considerablemente [13]. Adicionalmente, las simulaciones fueron particionadas en dos niveles de forma estática, y esta partición de tareas fue adecuada para la cantidad de simulaciones y pruebas que se realizaron, pero puede no ser adecuada si la cantidad de simulaciones cambia, en cuyo caso los desarrolladores deberán modificar de forma manual la estrategia de gridificación.

Idealmente, la Computación Grid debe ofrecer a los usuarios una forma fácil de programar sus aplicaciones para que luego sean desplegadas dentro de una Grid para localizar y utilizar los recursos necesarios para ejecutar las aplicaciones de forma eficiente. Sin embargo, hasta la actualidad este objetivo no ha sido cumplido por completo [16]. Esto se debe a que "gridificar" una aplicación requiere que los desarrolladores reescriban el código de la misma para utilizar los servicios provistos por la Grid subyacente previo a aprovechar sus recursos [13]. Así, la posibilidad de explotar Grids queda restringida a usuarios que posean un sólido conocimiento de estas tecnologías.

Surge así pocos años atrás una línea de investigación que busca disminuir los conocimientos de Computación Grid y particularmente paralelización que deben poseer los desarrolladores para aprovechar los recursos provistos. Como resultado, las *herramientas de gridificación* [13] posibilitan el despliegue de aplicaciones en Grids de forma más sencilla aislando al programador de las complejidades de la Grid subyacente. Estas herramientas se encuentran divididas en dos categorías. Por un lado se encuentran las basadas en la idea de API que se utilizan como interfaz en la comunicación de dos componentes de software, o en este caso la componente de software que contiene la lógica del negocio y el/los componente/s que engloba/n a la Grid subyacente y sus servicios. Ejemplos son JavaSymphony [7] y Java CoG Kit [10]. La utilización de estas APIs implica la ubicación, en el código de la lógica negocio, de llamadas a los métodos API que finalmente acceden a los servicios de los componentes del middleware (servicios) que gestionan los recursos de la Grid [14]. Por otro lado, y con el objetivo de abstraer aún más al desarrollador, se encuentran alternativamente los métodos de gridificación [13], que permiten paralelizar porciones de una aplicación, de manera (semi) automática, para obtener la contraparte Grid capaz de aprovechar los recursos ofrecidos de una Grid con una mínima intervención del desarrollador, y hasta en algunos casos, sin introducir código referido a Computación Grid dentro de la lógica de la aplicación.

3. Objetivos

Por lo expuesto, el plan de trabajo tiene por objetivo desarrollar una herramienta de gridificación que permita asistir a la gridificación de aplicaciones de simulación agropecuaria sin mezclar el código de la Grid subyacente con el código de la lógica de la aplicación fomentando la legibilidad, mantenibilidad y portabilidad sin afectar de forma considerable el desempeño de la aplicación respecto de esquemas de gridificación manuales. Se explorará e implementará paralelismo utilizando un híbrido de paralelización Workflow y el concepto de Dataflow [22, 6]. Para ello, se deberá definir las dependencias de cómputo entre tareas del sistema de simulación a gridificar, como así también las dependencias de datos (esto es, porciones de datos de salida computadas por una tarea necesarias para el comienzo de otra tarea), y las porciones específicas que las hacen dependientes. Esta definición se utilizará como base para paralelizar los componentes de interés del sistema objetivo, de manera de facilitar la transformación de código de simulación a su contraparte gridificada.

Como objeto de estudio se utilizará Simugan [12], un simulador diario ganadero Web [12] desarrollado en la Facultad de Ciencias Veterinarias de la UNICEN mediante la utilización de métodos ágiles [3] que ha sido exitosamente utilizado durante los últimos 10 años [11], y en la actualidad presenta más de 1000 horas de desarrollo. El simulador contiene modelos bioinformáticos que interactúan para representar la biología de base. Además, el simulador posee reglas de manejo empresarial para determinar distintas alternativas a aplicar sobre el establecimiento agropecuario. El simulador considera además aspectos sociales, debido a que el proceso de toma de decisiones es netamente humano y afectado no sólo por variables productivas, sino también por características personales del decisor [17]. Adicionalmente, el simulador brinda la posibilidad de utilización por múltiples usuarios y la posibilidad de realizar réplicas experimentales en base a variabilidad climática o precios, entre otras. Actualmente, este simulador se está usando en cursos de grado y postgrado de la Facultad de Ciencia Veterinarias de la UNICEN y el INTA, pero se pretende que este simulador sea utilizado por cátedras de grado y de postgrado de las facultades de Agronomía, Veterinarias y Zootecnia de todo el país, que en total son 38 unidades académicas.

Para la evaluación de la herramienta de gridificación se diseñará un conjunto de experimentos que buscarán obtener el tiempo de ejecución promedio y el factor speedup al gridificar Simugan tanto en baja carga como

en carga alta. La carga está definida como la cantidad de simulaciones ejecutándose al mismo tiempo o de forma concurrente. A fin de comparar alternativas de paralelización, las simulaciones se ejecutarán utilizando un enfoque secuencial (sin paralelismo), versus paralelizadas a nivel simulación, paralelizadas a nivel de tarea (Workflow), y finalmente paralelizadas a nivel de dependencia de datos entre tareas (Dataflow). La paralelización a nivel de simulación significa que las simulaciones de varios usuarios ejecutarán de forma concurrente como cajas negras. La paralelización a nivel de tarea significa que las simulaciones y las tareas no dependientes se ejecutarán de forma concurrente. Por último, la paralelización a nivel de dependencia de datos entre tareas significa que las simulaciones y todas las tareas (incluso las dependientes) se ejecutarán de forma concurrente, y una tarea entrará en espera cuando necesite alguna salida producida por una tarea de la cual depende. Para la realización de estas pruebas se tiene montado un Grid de 70 cores corriendo GridGain 5.2[1] como middleware Grid.

4. Cronograma

Análisis: Se estudiará la arquitectura de componentes que exhibe el simulador Simugan y la interacción entre éstos durante la utilización de distintas funcionalidades. Posteriormente, se analizará qué tipo de componentes pueden ser susceptibles de distribución de forma tal de permitir que el simulador aproveche varios procesadores desde el punto de vista de la estructura interna del mismo. Estas tareas se apoyarán en avances ya producidos en el contexto de investigaciones del grupo director [4].

Desarrollo: Se llevará a cabo el diseño y la implementación de la herramienta de gridificación que será propuesta como solución al problema planteado, haciendo énfasis en los objetivos de calidad descritos más arriba.

Experimentación: Se evaluará dicha herramienta de gridificación analizando los tiempos de ejecución promedio y el factor de speedup resultante de gridificar Simugan, bajo distintos escenarios representativos, en un Grid real de 70 cores.

Resultado: Finalmente, se procederá a la escritura del informe final de tesis. En términos de software, un resultado importante será la implementación paralela propiamente dicha de Simugan luego de aplicar la herramienta de gridificación.

El tiempo total se ha estipulado en 8 (ocho) meses, pero vale la pena aclarar que varias actividades han sido finalizadas o poseen avances importantes, con lo que se estima que el tiempo restante para la finalización del trabajo es de 3 (tres) a 4 (cuatro) meses.

Referencias

[1] Lajpat R. Ahuja, Liwang Ma, and Dennis J. Timlin. Trans-disciplinary soil physics research critical to synthesis and modeling of agricultural systems. *Soil Science Society American Journal*, 70:311–326, 2006.

[2] Mauricio Arroqui. Un método de gridificación para aplicaciones de simulación agropecuaria. Technical report, Universidad Nacional del Centro de la Provincia de Buenos Aires, 2013.

[3] Mauricio Arroqui, Pablo Mangudo, Claudia Marcos, and Claudio Machado. Agile development for a beef-cattle farm simulator. *IEEE Latin American Transactions*, Vol 7, issue 5:578–585, 2009.

[4] Mauricio Arroqui, Cristian Mateos, Claudio Machado, and Alejandro Zunino. Restful web services improve the efficiency of data transfer of a whole-farm simulator accessed by android smartphones. *Computers and Electronics in Agriculture*, 87(0):14 – 18, 2012.

[1] `http://www.gridgain.com`

[5] J. Aryal, D. Kulasiri, and D. Liu. Optimisation approaches for a complex dairy farm simulation model. *International Journal of Computer, Information and System Science, and Engineering*, 2:156–162, 2008.

[6] George Bosilca, Aurelien Bouteiller, Anthony Danalis, Thomas Herault, Pierre Lemarinier, and Jack Dongarra. Dague: A generic distributed dag engine for high performance computing. *Parallel Computing*, 38:37–51, 2012.

[7] Thomas Fahringer and Alexandru Jugravu. Javasymphony: a new programming paradigm to control and synchronize locality, parallelism and load balancing for parallel and distributed computing: Research articles. *Concurr. Comput. : Pract. Exper.*, 17(7-8):1005–1025, June 2005.

[8] Ian Foster and Carl Kesselman. *The Grid 2: : Blueprint for a New Computing Infrastructure*. The Elsevier Series in Grid Computing, 2003.

[9] B.A Keating, P.S Carberry, G.L Hammer, M.E Probert, M.J Robertson, D Holzworth, N.I Huth, J.N.G Hargreaves, H Meinke, Z Hochman, G McLean, K Verburg, V Snow, J.P Dimes, M Silburn, E Wang, S Brown, K.L Bristow, S Asseng, S Chapman, R.L McCown, D.M Freebairn, and C.J Smith. An overview of apsim, a model designed for farming systems simulation. *European Journal of Agronomy*, 18(3 - 4):267 – 288, 2003.

[10] Gregor Von Laszewski, Jarek Gawor, Peter Lane, Nell Rehn, and Mike Russell. Features of java commodity grid kit. *Concurrency and Computation: Practice and Experience*, 14 (13-15):1045–1055, 2003.

[11] C. F. Machado, J. C. Burges, H. Berger, C. Faverón, and I. Steffanazzi. *First steps in the use of a web whole-farm model to foster the feedback between beef cattle extension and research.* An Overview of Research on Pastoral-Based Systems in South America, 2010.

[12] Claudio Machado, Steve Morris, John Hodgson, Mauricio Arroqui, and Pablo Mangudo. A web-based model for simulating whole-farm beef cattle systems. *Computers and Electronics in Agriculture*, 74(1):129 – 136, 2010.

[13] Cristian Mateos, Alejandro Zunino, and Marcelo Campo. A survey on approaches to gridification. *Software: Practice and Experience*, 38:523–556, 2008.

[14] Cristian Mateos, Alejandro Zunino, and Marcelo Campo. Grid-enabling applications with jgrim. *International Journal of Grid and High Performance Computing*, 1(3):52–72, 2009.

[15] Cristian Mateos, Alejandro Zunino, and Marcelo Campo. An approach for non-intrusively adding malleable fork/join parallelism into ordinary javabean compliant applications. *Computer Languages, Systems & Structures*, 36(3):288 – 315, 2010.

[16] Cristian Mateos, Alejandro Zunino, Matías Hirsch, and Mariano Fernández. Enhancing the byg gridification tool with state-of-the-art grid scheduling mechanisms and explicit tuning support. *Advances in Engineering Software*, 43(1):27 – 43, 2012.

[17] R. L. McCown. Changing systems for supporting farmer's decisions: problems, paradigms, and prospects. *Agricultural Systems*, 74:179–220, 2002.

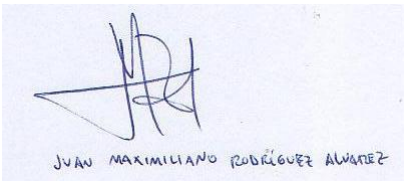
[18] Alvaro J. Romera, Steve T. Morris, John Hudgson, W. D. Stirling, and S. J. R. Woodward. The influence of replacement policies on stability of production in a simulated cow-calf farm system. *New Zealand Journal of Agricultural Research*, 49:1:35–44, 2006.

[19] Rob V. Van Nieuwpoort, Gosia Wrzesińska, Ceriel J. H. Jacobs, and Henri E. Bal. Satin: A high-level and efficient grid programming model. *ACM Trans. Program. Lang. Syst.*, 32(3):9:1–9:39, March 2010.

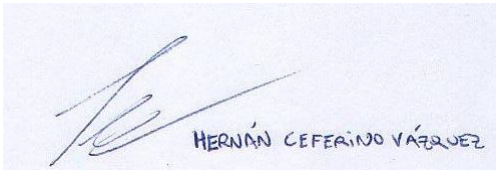
[20] S. R. J. Woodward, A. Romera, W. Beskow, and S. J. Lovatt. Better simulation modelling to support farming system innovation: review and sysnthesis. *New Zealand Journal of Agricultural Research*, 51:235–252, 2008.

[21] Gang Zhao, Brett A. Bryan, Darran King, Zhongkui Luo, Enli Wang, Ulrike Bende-Michl, Xiaodong Song, and Qiang Yu. Large-scale, high-resolution agricultural systems modeling using a hybrid approach combining grid computing and parallel processing. *Environmental Modelling and Software*, 2012.

[22] Daniel Zinn, Shawn Bowers, Sven Köhler, and Bertram Ludäscher. Parallelizing xml data-streaming workflows via MapReduce. *Journal of Computer and System Sciences*, 76(6):447 – 463, 2010.

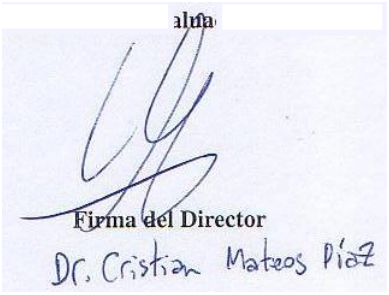


Firma del alumno 1



Firma del alumno 2

Avalo la presente solicitud de evaluación,



Firma del Director

Firma del Co-director