Modelización no lineal del problema de la distribución de la caña de azúcar y su implementación en Sistemas Multiagentes Organizacionales

Nicolás Majorel Padilla, Pedro Araujo, Adrian Will y Sebastián Rodríguez
Grupo de Investigación en Tecnologías Informáticas Avanzadas
Universidad Tecnológica Nacional - Facultad Regional Tucumán
Rivadavia 1050, Tucumán, Argentina
{*nicolas.majorel,pedro.araujo,adrian.will,sebastian.rodriguez*}*@gitia.org*

Abstract

La industria del azúcar de caña en Tucumán es de las actividades económicas más importantes de la provincia, pero a pesar de ello su distribución y logística no presenta una planificación optimizada. Este trabajo aborda dicha problemática a través de dos objetivos: en primer lugar presentar una mejora de modelos matemáticos previos, incorporando restricciones adicionales y factores no lineales; y por otro lado, mostrar una solución desarrollada utilizando el paradigma de Sistemas Multiagentes basado en el enfoque Organizacional, haciendo uso de las técnicas y herramientas adecuadas.

La modelización incluye una variante original del Generalized Assignment Problem y un nuevo conjunto de pruebas necesarias para establecer valores de referencia. El uso del enfoque organizacional derivó en diversas ventajas que permiten fácilmente escalar el problema.

1. Introducción

El presente trabajo tiene su origen en el análisis de la distribución de la caña de azúcar en la provincia de Tucumán, Argentina, desde las parcelas donde es producida hasta los ingenios donde es procesada. A pesar de ser una de las industrias más importantes de la provincia, la caña de azúcar carece de una planificación adecuada de toda su cadena productiva. Este hecho no solamente reduce los beneficios de los productores de caña y de los ingenios, sino que también produce un importante impacto en la congestión de rutas, accidentes viales y contaminación ambiental. Pueden encontrarse muchos esfuerzos de optimización de problemas similares en otros países [25][15][16][36], pero dadas las diferencias sustanciales entre las cadenas productivas de los distintos países, las soluciones propuestas no son directamente aplicables en el escenario planteado.

En trabajos previos presentados por los autores [21] [11] se trabajó con versiones preliminares de este problema, y se desarrolló un modelo matemático y diferentes algoritmos heurísticos para resolverlo. En todos los casos, los mejores resultados se obtuvieron con heurísticas basadas en Sistemas Multiagentes (SMA de ahora en adelante). En el presente trabajo, se presenta un modelo matemático más avanzado que tiene en cuenta mayor cantidad de restricciones y aspectos del problema. El modelo resultante es una variante no lineal del conocido problema *General Assignment Problem*. Ante este nuevo enunciado y en vista de los resultados anteriores, se decidió continuar trabajando con Sistemas Multiagentes, mejorando la modelización del problema utilizando mejores herramientas de Sistemas Multiagentes que conllevan una serie de ventajas que consideramos significativas tales como la autonomía y la escalabilidad.

El resto del artículo está organizado de la siguiente manera: en la sección 2 se describe el problema planteado, comenzando por sus características generales, detallando problemas similares y presentando su formulación matemática. Luego, en la sección 3 se define un conjunto de pruebas para evaluarlo, y en la sección 4 se detalla el modelado organizacional y la implementación con SMA. Por último, se presentan las conclusiones y trabajos futuros en la sección 5.

2. Descripción del problema

En esta sección se describirá el problema que se intenta resolver, comenzando por las características generales que presenta la cosecha de la caña de azúcar en la provincia de Tucumán, continuando por la descripción de problemas similares y las diferencias con el problema actual, y finalizando con el modelo matemático completo que lo describe.

2.1. Características generales

En el caso planteado, la eficiencia de una fábrica (ingenio) depende de manera no lineal del régimen de trabajo a la que se ve sometida. En esos casos, la producción del ingenio con una carga de trabajo mínima o máxima resulta ser ineficiente. Desde un punto de vista global, la mejor distribución de carga de trabajo en términos de eficiencia depende de diversos factores, como ser las características del propio ingenio, la época del año considerada, la calidad de la materia prima, entre otros. Para el modelo que se propone, los más importantes son los siguientes:

- Existe una capacidad mínima que debe ser satisfecha para cada ingenio, por debajo de la cual el ingenio no puede trabajar y debe detener su producción. El inicio de la producción es un proceso muy costoso, y en condiciones normales ocurre solamente una vez por año.
- Existen deterioros o pérdidas de materia prima en los ingenios, usualmente asociados a problemas de estacionamiento (largas esperas antes de que sea procesada), y por mal estado de las rutas.
- La mayoría de los ingenios en Tucumán alcanzan su rendimiento pico al trabajar a un determinado porcentaje de su capacidad máxima. Resultados teóricos establecen que este porcentaje suele ser entre $60\,\%$ y $95\,\%$, que depende de cada ingenio, y que la eficiencia disminuye al trabajar por encima o por debajo del mismo.[1]
- El modelo se limita a manejar la distribución de la producción de un solo día, entendiendo que es el caso más importante. En este caso, se estima que en promedio se cosechan 200 parcelas por día, y que en plena producción están trabajando los 15 ingenios de la provincia, haciendo un total de 15^{200} combinaciones posibles.

2.2. Problemas similares

Dada las características del problema mencionadas, puede asemejarse al *Generalized Assignment Problem (GAP)*, un problema muy conocido dentro de la investigación operativa y la optimización lineal. El GAP es definido en [8] como "un problema de optimización combinatoria bien conocido, NP-completo, que consiste en encontrar la asignación de tareas a agentes que genere mínimos costos, de manera que cada tarea es asignada a exactamente un agente, sujeto a la capacidad de dicho agente" . Pueden encontrarse diversas aplicaciones de este problema, en áreas como logística, distribución o problemas de enrutamiento de vehículos, por mencionar algunos. Una recopilación de aplicaciones del GAP puede encontrarse en [6], y diversas extensiones y generalizaciones del mismo fueron recopiladas en [31].

Para expresar matemáticamente el GAP, se usará la notación empleada en [24]: n representa el número de agentes (ingenios) y m el número de tareas (parcelas) que pueden ser asignadas a los agentes. Cuando una parcela i es asignada al ingenio j se expresa como x_{ij}, y se genera un costo c_{ij} o un beneficio p_{ij} asociado a dicha asignación. Además, cada ingenio tiene una capacidad máxima b_j, y una parcela i le aporta una cantidad de producción expresada como a_{ij}. El GAP busca encontrar una asignación de todas las parcelas en los ingenios, de manera que la suma de las producciones aportadas por todas las parcelas asignadas a un ingenio no exceda la capacidad máxima del mismo, y el costo total de la asignación sea mínimo, manteniendo siempre que cada parcela debe ser asignada a un único ingenio.

El GAP es un problema NP-hard [13] y también APX-hard [7]. Inclusive, el problema de saber si existe una solución válida es NP-completo[22]. Muchos algoritmos determinísticos y heurísticos fueron desarrolados para el GAP, como puede verse en [6] [22] [23] [8] [20] [26] [35] [32] [24], y también muchas variantes. Entre estas variantes queremos destacar el Minimum Quantity GAP [18] donde la capacidad de un ingenio tiene una cota mínima, el Multi-resource GAP [20] [33] en el que más de una recurso es asignado a cada ingenio, el Additive Non-Linear GAP [29] en el que se añade un término no lineal a la función, entre otros [5] [34] [10] [28] [19].

2.3. Modelo matemático

A pesar de todas las variantes del GAP analizadas, ninguna es suficiente para modelar correctamente el problema de la manera en la que fue propuesto. Es imprescindible que el modelo considere la capacidad mínima de los ingenios y la curva no lineal de eficiencia. Por lo tanto, proponemos una nueva variante del GAP que se formula de la siguiente manera:

$$\text{maximizar } Z(X) = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}x_{ij}h_j\left(\sum_{i=1}^{n} a_{ij}x_{ij}\right) \quad (1)$$

sujeto a:

$$d_j \leq \sum_{i=1}^{n} a_{ij}x_{ij} \leq b_j \quad j = 1,\ldots,m \quad (2)$$

$$\sum_{j=1}^{m} x_{ij} = 1 \quad i = 1,\ldots,n \quad (3)$$

$$x_{ij} = \begin{cases} 1, & \text{si parcela i es asignada a ingenio j} \\ 0, & \text{en otro caso.} \end{cases} \quad (4)$$

En las ecuaciones anteriores se observa que:

[1] La determinación de la eficiencia de un ingenio es un problema aparte, de otra área del conocimiento, y fuera de los alcances de este trabajo.

- Z es la eficiencia global del sistema.
- $x_{ij} = 1$ si la parcela i entrega su producción al ingenio j, y 0 si no lo hace.
- a_{ij} es la producción en toneladas que la parcela i entrega al ingenio j. Depende del ingenio j para modelar las pérdidas ocasionadas por el ingenio mismo o por las rutas.
- d_j es la capacidad mínima que necesita el ingenio para mantener su producción.
- Las ecuaciones 3 y 4 establecen que cada parcela entrega toda su producción a exactamente un ingenio.
- h_j representa la eficiencia a la cual se encuentra trabajando el ingenio j, en función del porcentaje de su capacidad máxima. Es una función que depende de la solución completa, debido a que si se modifica alguna asignación para que la producción de una parcela determinada cambie del ingenio j_1 al ingenio j_2, se afecta la eficiencia de ambos ingenios, y en algunos casos de manera significativa, dependiendo del tamaño de a_i relativo a las capacidades b_{j1} y b_{j2}.

La curva de eficiencia versus capacidad de un ingenio dado puede ser cualquier función que cumpla con las siguientes propiedades: (a) alcanza un pico dentro del rango de capacidades válidas (este pico suele ser alrededor del $85\,\%$ de la capacidad máxima); (b) es monótona creciente hasta ese pico, y monótona decreciente después del mismo; (c) la convexidad puede cambiar en varios puntos de la función; y (d) sus valores son números reales entre 0 y 1. Bajo estas condiciones, la curva de eficiencia puede tomar diferentes formas, como se muestra en la Figura 1.

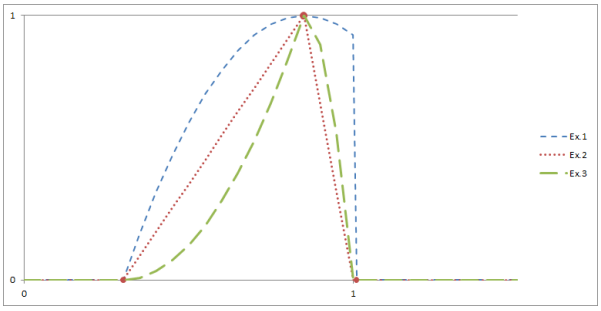

Figura 1. Ejemplos de curvas de eficiencia de un ingenio en función del porcentaje utilizado de la capacidad máxima.

3. Generación de un banco de pruebas

En la sección anterior se introdujo una nueva variante del GAP, y para poder obtener resultados sobre esta nueva variante, es necesario proveer un conjunto de pruebas tipo benchmark. En la literatura puede encontrarse una suite estándar para el GAP, que consiste en cinco tipos de instancias diferentes denominadas Tipos A, B, C, D y E, como fueron definidas en [8, 35]. Los Tipos D y E son más difíciles de resolver que los demás porque c_{ij} y a_{ij} son inversamente proporcionales. La suite mencionada consiste en 57 instancias en total, con m hasta 1600. Todas estas instancias están disponibles a través de la OR-Library [4] y de [30], pero sus resultados no pueden ser usados como referencia para la variante propuesta porque fueron generados para el GAP, y por tanto no consideran el factor h_j no lineal introducido. Otras variantes del GAP no presentaron instancias para realizar pruebas [29] [10], o generaron sus propias instancias que no están disponibles [28]. Y tampoco es posible utilizar CPLEX, GAMS, o programas similares porque la variante no es lineal.

Entonces, para poder evaluar la variante presentada del GAP fue necesario desarrollar un conjunto de instancias idénticas a las descriptas pero de menor tamaño. Fueron seleccionadas 20 instancias diferentes de complejidad media que resultan representativas del conjunto total, y las mismas están disponibles en [2]. Estas instancias fueron sometidas en primer lugar a un algoritmo de fuerza bruta desarrollado en lenguaje Julia [1], con el único fin de obtener los valores óptimos como referencias para probar heurísticas. Se decidió utilizar Julia como lenguaje de desarrollo de este algoritmo por dos motivos principales: su sencillez para programar, al ser muy similar a MatLab, y por presentar una performance mucho mayor que este, comparable con algoritmos desarrollados en C/C++. En este algoritmo, la función h_j utilizada fue una curva gaussiana con una media en $0{,}85$ y una desviación estándar de $0{,}25$ definida para el intervalo [0.30, 1.00], y con el valor cero fuera de tal intervalo. Esta función fue seleccionada porque cumple con todas las propiedades que debe tener la curva de eficiencia de los ingenios.

Los valores óptimos obtenidos para estas instancias pueden observarse en la Tabla 1 en la página 4. En dicha tabla, la columna etiquetada “Combinaciones” muestra el número de todas las posibles soluciones, incluso aquellas que no satisfacen las restricciones, mientras que la columna etiquetada “Soluciones” muestra únicamente las soluciones válidas. La columna “Válidas” muestra el cociente entre las dos columnas previas, y sirve como indicador de la dificultad de la instancia. Finalmente, se muestra el valor óptimo Z obtenido.

Type	m	n	Combinaciones	Soluciones	Válidas	OptValue
A	3	12	531441	316307	59.52 %	0.658656
A	4	10	1048576	145409	13.87 %	0.497642
A	5	10	9765625	884643	9.06 %	0.544364
A	3	15	14348907	8097146	56.43 %	0.871947
A	4	12	16777216	4596196	27.40 %	0.628855
B	3	12	531441	27992	5.27 %	0.543980
B	4	10	1048576	3700	0.353 %	0.614256
B	5	10	9765625	1570	0.0161 %	0.542515
B	3	15	14348907	2632	0.0183 %	0.675879
B	4	12	16777216	4992	0.0298 %	0.606471
C	3	12	531441	104	0.0196 %	0.564030
C	4	10	1048576	23	0.00219 %	0.459722
C	5	10	9765625	102	0.00104 %	0.473988
C	3	15	14348907	64	0.00045 %	0.535528
C	4	12	16777216	89	0.00053 %	0.501473
D	3	12	531441	250	0.0470 %	0.277288
D	4	10	1048576	405	0.0386 %	0.284239
D	5	10	9765625	1500	0.0154 %	1.106979
D	3	15	14348907	2306	0.0161 %	0.308865
D	4	12	16777216	9859	0.0588 %	0.284524

Tabla 1. Benchmark de referencia para las instancias seleccionadas.

4. Implementación con SMA

En esta sección se describirá la implementación de la solución utilizando SMA. Primero se describen las características del enfoque organizacional y las herramientas utilizadas, y luego la manera en la que este modelado es utilizado para generar una solución al problema propuesto.

4.1. Enfoque organizacional para SMA

Los SMA han demostrado en las últimas décadas su enorme potencial abordando una amplia gama de problemas que va desde la robótica hasta la resolución de problemas distribuidos, entre otros. El agente es el componente principal del paradigma, puede ser considerado como una entidad física o virtual con un alto grado de autonomía, independiente, capaz de cooperar, competir, comunicarse, actuar flexiblemente y ejercer control sobre su comportamiento dentro del marco de sus objetivos. Estas entidades evolucionan dentro de un ambiente al cual son capaces de percibir y modificar. En contraste, un sistema multiagentes está caracterizado por múltiples agentes inteligentes que interactúan entre sí para alcanzar objetivos que pueden ser compartidos o no entre los agentes.

El campo de los SMA es muy prolífico y en constante evolución. En el mismo se han propuesto numerosos enfoques, entre los cuales el organizacional ha cobrado especial importancia. Inspirado en la metáfora social, es usado en la definición tanto de metodologías como metamodelos. Este novedoso enfoque es una evolución de la forma de modelar que parte de una visión donde el sistema centra su atención principalmente en el agente y sus aspectos individuales a una visión donde el sistema es considerado como una organización en el cual los agentes forman grupos y jerarquías además de seguir reglas y comportamientos específicos.

Ferber [12] propone un conjunto de principios generales que deben ser tomados en cuenta cuando se hace uso de este enfoque: (i) El nivel organizacional describe el "qué" y no el "cómo". En otras palabras, el nivel organizacional impone un patrón de actividades en los agentes, pero no describe como éstos deben comportarse. (ii) No se realiza ninguna descripción de los agentes por lo que no existen potenciales conflictos sobre los problemas mentales a nivel organizacional. (iii) Una organización provee una forma de particionar un sistema, en donde cada partición (o grupo de agentes) constituye un contexto de interacción. Así, un grupo es una unidad organizacional en el cual los miembros son capaces de interactuar libremente.

A diferencia de lo realizado en los artículos anteriores [21], en el presente trabajo utilizaremos herramientas propias de la teoría de agentes para permitir en primer término capitalizar las ventajas que ofrecen tales herramientas y evitar, a su vez, las potenciales divergencias implementatorias que pueden surgir en la utilización de un paradigma distinto al de agentes, como puede ser el del paradigma objetos.

El paradigma adoptado para la implementación de la

lógica del problema descripto en las secciones anteriores es un SMA basado en el enfoque Organizacional. Para el mismo se utilizó el *ecosistema* creado alrededor de la metodología ASPECS [9]; conformado por el lenguaje de modelado CRIO [27], la herramienta de desarrollo Janeiro Studio [3] y la plataforma de ejecución Janus [14]. Dicho proceso es considerado en la actualidad como uno de las más completos dentro de la literatura de agentes [17].

El metamodelo CRIO es la base fundamental del proceso ASPECS. Este metamodelo debe su nombre al acrónimo formado por sus cuatro conceptos principales: *Capacidad*, descripción de un *know-how*/servicio; *Rol*, es el comportamiento esperado y el conjunto de derechos y obligaciones dentro de la organización; *Interacción*, secuencia de eventos intercambiadas entre roles; y por último, *Organización*, definida como una colección de roles y sus interacciones dentro de un determinado contextos.

En la Figura 2 se ilustra el diagrama organizacional para el problema, modelado mediante Janeiro Studio. Esta herramienta facilita la manipulación de las instancias del metamodelo adaptado mientras que provee lineamientos para un correcto uso de la metodología ASPECS. Janeiro Studio provee soporte a los cinco diagramas principales que contemplan la primera fase de la metodología (fase de *dominio del problema*): Diagrama de Requerimientos del Dominio, Diagrama de Ontología, Diagrama Organizacional, Diagrama de Interacción y Diagrama de Comportamiento.

En la Figura mencionada puede observarse a la organización Zafra compuesta por dos roles: *Consumidor* (Ingenio) y *Productor* (Parcela), ambos relacionados a través de un protocolo (círculo gris). Además, los roles tienen asociadas capacidades: *Consumidor* requiere de las capacidades *CalcularRendNorm* (Abreviación de Calcular Rendimiento Normalizado), *CalcularFuerza* y *CalcularAcumulador*, mientras que el rol *Productor* sólo tiene asociada la capacidad *ElegirConsumidor*. La finalidad de cada una de estas capacidades serán descriptas más adelante.

Los modelos generados luego se convertirán en códigos en lenguaje Java que serán ejecutados en la plataforma Janus. Janus es una plataforma para SMA implementada en Java 1.6 que permite a los desarrolladores crear aplicaciones basadas en SMA de manera eficiente y veloz, proveyendo un conjunto de características para desarrollar, ejecutar y mostrar tales aplicaciones, como así también monitorearlas y permitir que las mismas puedan distribuir su ejecución en una red de computadoras.

Es importante destacar en este punto que todas las herramientas utilizadas son gratuitas, de código abierto y de libre distribución.

4.2. Solución del problema planteado

En esta sección se describirá el funcionamiento del SMA. El Diagrama de Interacción que muestra la secuencia sistemática de intercambio de mensajes entre los roles puede observarse en la Figura 3.

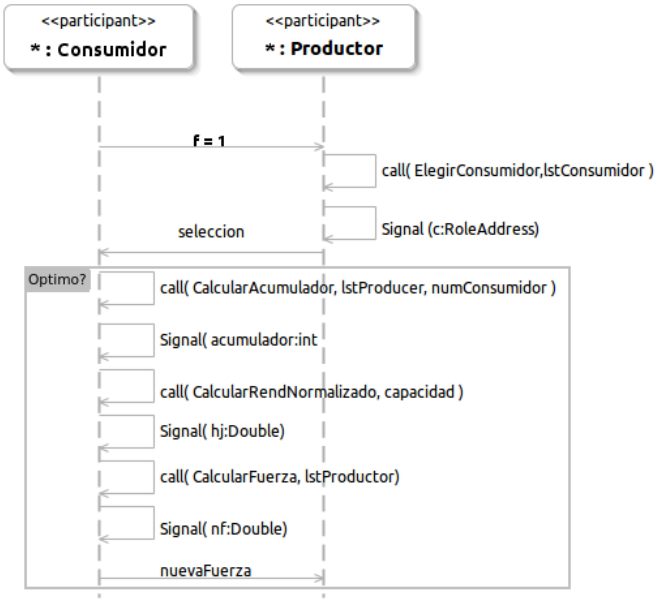

Figura 3. Diagrama de Interacción.

Inicialmente los agentes que jueguen el rol *Consumidor* establecen sus fuerzas f iniciales en 1 para atraer a los agentes que juegan el rol *Productor* con la misma intensidad evitando sesgos. Adicionalmente todos los consumidores mantienen dos listados como parte de los estados mentales de los agentes: Un listado de los productores que le fueron asignados y uno de los que no le fueron asignados. En esta inicialización, todos los productores por defectos incluidos dentro del listado de productores no asignados de todos los consumidores. La simulación en Janus comienza cuando cada uno de los consumidores envían, mediante un mensaje del tipo broadcast, las fuerzas iniciales (f=1) a los productores. A continuación, los agentes que juegan este rol se bloquean en espera de una respuesta por parte de los productores. A medida que los productores reciben las fuerzas iniciales, evalúan la preferencia que pueden tener por un consumidor en particular. Dicho cálculo surge del cociente entre la fuerza y el costo que tiene el productor para ese consumidor (C_{ij}). Para lograr el objetivo descripto, el rol realiza un llamado a su capacidad *ElegirConsumidor* pasando como parámetro la lista de consumidores. Terminada la evaluación y establecido cuál es el consumidor seleccionado, el agente dispara una señal indicando a su rol que finalizó la actividad. Esta señal devuelve la dirección del consumidor

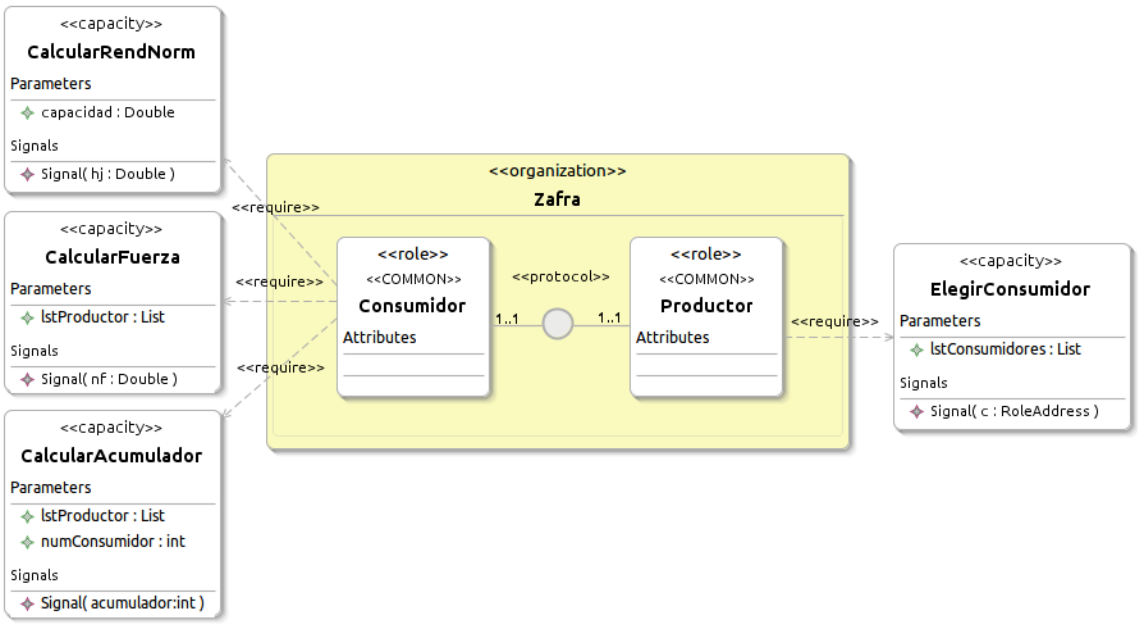

Figura 2. Diagrama Organizacional del problema.

que será utilizada para notificar a éste que fue elegido.

Cuando un consumidor recibe un mensaje que le indica que fue seleccionado por un productor, actualiza su listados de productores, moviendo al productor emisor del mensaje del listado de productores no asignados al de productores asignados (*lstProductor*). La primera capacidad requerida es *CalcularAcumulador* que calcula cuan lleno está el consumidor con respecto a su capacidad máxima. Idealmente este es un valor comprendido entre 0 y 1, y aunque temporalmente puede ser mayor que 1, indicando que el consumidor está excedido en su capacidad máxima, nunca será menor que 0 ni es permitido que sea mayor que 1 en la solución final debido a restricciones operativas.

Una vez que la capacidad *CalcularAcumulador* devuelve su resultado, el agente llama a la capacidad *CalcularRendNorm* que determina el factor de rendimiento h_j. En el caso del modelo planteado, la función es una curva gaussiana tal como se explicó al comienzo del presente trabajo, con media en $0{,}85$ y desviación estándar $0{,}25$, entre la capacidad mínima de los consumidor (usualmente $0{,}30$) y la capacidad máxima ($1{,}00$). Fuera de este intervalo, el rendimiento es 0. Finalizada la tarea, la capacidad retorna la variable h_j que será utilizada por el rol para determinar su aporte a la energía total del sistema. En el caso del algoritmo propuesto este valor resulta ser el valor de h_j, dividido por el costo correspondiente a C_{ij}.

Finalmente, cada consumidor recalcula -como parte de la ejecución de su última capacidad (*CalcularFuerza*)- la fuerza f que le enviará a cada productor i según las siguientes estrategias: para los productores asignados, si su capacidad utilizada es mayor que su capacidad óptima entonces intentará deshacerse de algunos productores, disminuyéndoles la fuerza correspondiente; en cambio, si su capacidad utilizada no es mayor que su capacidad óptima entonces mantiene su valor de fuerza actual. Ahora para el conjunto de consumidores ausentes, si su capacidad utilizada es menor que su capacidad óptima entonces intentará atraer algunos productores, aumentando la fuerza correspondiente. En el caso de que su capacidad no sea menor que su capacidad óptima, entonces disminuirá levemente su fuerza para evitar atraer nuevos productores. Una vez calculada la nueva fuerza, esta es enviada a los productores repitiendo el mismo ciclo hasta que cumpla una condición de terminación.

En cuanto a la condición de terminación, en la mayoría de los casos el sistema alcanza un estado de equilibrio en el punto de menor energía, donde todos los productores y consumidores están asignados. Sin embargo, existen algunos ejemplos de ciclo límite y otros casos en estudio donde un cierto número de productores itera en un ciclo repetitivo. En este caso dichos productores, movilizados por el cambio de fuerza de los consumidores, se mueven periódicamente alrededor de un grupo fijo de consumidores. Sin embargo el cambio en energía, y otras variables, es poco significativo.

5. Conclusiones y Trabajos Futuros

En este trabajo se introdujo una nueva modelización matemática para el problema de la distribución de la caña de azúcar desde las parcelas hasta los ingenios en la provincia

de Tucumán. Esta modelización incluye una variante original para un problema bien conocido como es el GAP.

El modelo matemático planteado contempla diversas mejoras sobre modelos anteriores, como costos diferentes de cada parcela hacia cada ingenio, y la eficiencia de cada ingenio en el procesamiento de la caña. Este modelo resulta notoriamente más complejo que otros presentados anteriormente para el mismo problema, dado que, entre otras razones, la eficiencia es no lineal como función del estado de funcionamiento del ingenio. Adicionalmente, se presentó un conjunto de pruebas para esta nueva variante.

Se presentó también un Sistema Multiagentes que implementa el algoritmo propuesto, modelizado en Janeiro Studio e implementado en la plataforma Janus. Ambas herramientas están especialmente diseñadas para trabajar con sistemas multiagentes. El modelado realizado fue basado en el enfoque organizacional, más precisamente en el propuesto por el proceso de desarrollo de ASPECS utilzando el lenguaje de modelado CRIO. La características del problema presentado a lo largo del artículo se ajusta a las características de este paradigma. Este enfoque presenta numerosas ventajas tales como la heterogenidad de los lenguajes, una alta modularidad, la posibilidad de definir múltiples arquiteturas de agentes, y otras. Además presenta una alta escalabilidad y adaptabilidad a nuevos requisitos, factores que resultarán críticos en cualquier implementación de este tipos de sistemas cuando la necesidad de simulación demande incrementar la cantidad de elementos.

En cuanto a líneas futuras de investigación, resulta importante ampliar el benchmark desarrollado para incluir ejemplos de mayor tamaño. También es necesario refinar la heurística presentada antes de intentar con pruebas piloto. En cuanto a la modelización organizacional se plantean varias posibilidades entre la que se destacan la necesidad de ejecutar la simulación en un entorno distribuido aprovechando las capacidades de la plataforma Janus, permitiendo disminuir los tiempos de ejecución y resolver así los casos de mayor tamaño.

6. Agradecimientos

Este trabajo fue parcialmente financiado por la Universidad Tecnológica Nacional UTN-PID 1318.

Referencias

[1] The julia language, 2012. http://julialang.org/.

[2] Benchmarks for vpgap, 2014. http://www.gitia.org/projects/vpgap/.

[3] Pedro Bernabé Araujo and Sebastián Alberto Rodríguez. Janeiro studio. *CONAIISI, Córdoba, Argentina*, 2013.

[4] John E Beasley. Or-library, 1990. http://people.brunel.ac.uk/~mastjjb/jeb/orlib/gapinfo.html.

[5] J. Geunes C. Rainwater and E.H. Romeijn. The generalized assignment problem with flexible jobs. *Discrete Applied Mathematics*, 157(1):49–67, 2009.

[6] Dirk G Cattrysse and Luk N Van Wassenhove. A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research*, 60(3):260–272, 1992.

[7] Chandra Chekuri and Sanjeev Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, 35(3):713–728, 2005.

[8] Paul C Chu and John E Beasley. A genetic algorithm for the generalised assignment problem. *Computers & Operations Research*, 24(1):17–23, 1997.

[9] Massimo Cossentino, Nicolas Gaud, Vincent Hilaire, Stphane Galland, and Abderrafia Koukam. Aspecs: an agent-oriented software process for engineering complex systems. *Autonomous Agents and Multi-Agent Systems*, 20(2):260–304, 2010.

[10] P. K. De and Bharti Yadav. A general approach for solving assignment problems involving withfuzzy cost coefficients". *Modern Applied Science*, 6 number 3:2–10, 2012.

[11] Agustín Décima, Nicolás Majorel Padilla, Adrián Will, Sebastián Rodríguez, and Oscar Diez. Optimizacion del transporte de caña de azúcar utilizando sistemas multiagentes y algortimos genéticos grouping. In *Mecánica Computacional, Volume XXX. Number 32. Industrial Applications (A)*, 2011.

[12] Jacques Ferber, Olivier Gutknecht, and Fabien Michel. From agents to organizations: An organizational view of multi-agent systems. In Paolo Giorgini, Jrg P. Mller, and James Odell, editors, *Agent-Oriented Software Engineering IV*, number 2935 in Lecture Notes in Computer Science, pages 214–230. Springer Berlin Heidelberg, 2004.

[13] Marshall L Fisher, Ramchandran Jaikumar, and Luk N Van Wassenhove. A multiplier adjustment method for the generalized assignment problem. *Management Science*, 32(9):1095–1103, 1986.

[14] Stphane Galland, Nicolas Gaud, Sebastian Rodriguez, and Vincent Hilaire. Janus: Another yet general-purpose multiagent platform. In *7th Agent-Oriented Software Engineering Technical Forum (TFGAOSE-10)*. Agent Technical Fora, 2010.

[15] AJ Higgins. Optimizing cane supply decisions within a sugar mill region. *Journal of Scheduling*, 2(5):229–244, 1999.

[16] Andrew J Higgins and Russell C Muchow. Assessing the potential benefits of alternative cane supply arrangements in the australian sugar industry. *Agricultural Systems*, 76(2):623–638, 2003.

[17] David Isern, David Snchez, and Antonio Moreno. Organizational structures supported by agent-oriented methodologies. *J. Syst. Softw.*, 84(2):169–184, 2011.

[18] Sven O Krumke and Clemens Thielen. The generalized assignment problem with minimum quantities. *European Journal of Operational Research*, 228(1):46–55, 2013.

[19] M.F. Monaco L. Moccia, J.F. Cordeau and M. Sammarra. A column generation heuristic for a dynamic generalized assignment problem. *Computers & Operations Research*, 36(9):2670–2681, 2009.

[20] Manuel Laguna, James P Kelly, JoséLuis González-Velarde, and Fred Glover. Tabu search for the multilevel generalized assignment problem. *European Journal of Operational Research*, 82(1):176–189, 1995.

[21] N. Majorel Padilla, A. Décima, A. Will, S. Rodriguez, and O. Diez. Optimization of sugar cane transportation in tucuman using multiagent systems. *Iberoamerican Journal of Industrial Engineering*, 3:103–119, 2011.

[22] Silvano Martello and Paolo Toth. *Knapsack problems*. Wiley New York, 1990.

[23] Ibrahim H Osman. Heuristics for the generalised assignment problem: simulated annealing and tabu search approaches. *Operations-Research-Spektrum*, 17(4):211–225, 1995.

[24] Lale Özbakir, Adil Baykasoğlu, and Pınar Tapkan. Bees algorithm for generalized assignment problem. *Applied Mathematics and Computation*, 215(11):3782–3795, 2010.

[25] LYNE PWL and BEZUIDENHOUT CN. The complexities of introducing the fredd vehicle scheduling system into the darnall mill area. In *Proc S Afr Sug Technol Ass*, volume 80, page 66. South African Sugar Technologists' Association, 2006.

[26] H Ramalhinho and Daniel Serra. Adaptive search heuristics for the generalized assignment problem. *Mathware & soft computing*, 9(3):209–234, 2008.

[27] Sebastian Rodriguez, Nicolas Gaud, Vincent Hilaire, Stphane Galland, and Abderrafia Koukam. An analysis and design concept for self-organization in holonic multi-agent systems. In Sven A. Brueckner, Salima Hassas, Mrk Jelasity, and Daniel Yamins, editors, *Engineering Self-Organising Systems*, number 4335 in Lecture Notes in Computer Science, pages 15–27. Springer Berlin Heidelberg, 2007.

[28] Kamran Shahanaghi, Hamidreza Haddad, Mohammad Hossein Babaie, and Camran Alavi. A new approach for solving the generalized assignment problem in uncertain environment. *Advances in Mathematical and Computational Methods*, 2(1):21–27, 2012.

[29] Thomas C Sharkey and H Edwin Romeijn. Greedy approaches for a class of nonlinear generalized assignment problems. *Discrete Applied Mathematics*, 158(5):559–572, 2010.

[30] M. Yagiura. Gap instances, 1998. `http://www.al.cm.is.nagoya-u.ac.jp/~yagiura/gap/`.

[31] Mutsunori Yagiura and Toshihide Ibaraki. The generalized assignment problem and its generalizations, 2007.

[32] Mutsunori Yagiura, Toshihide Ibaraki, and Fred Glover. A path relinking approach with ejection chains for the generalized assignment problem. *European journal of operational research*, 169(2):548–569, 2006.

[33] Mutsunori Yagiura, Shinji Iwasaki, Toshihide Ibaraki, and Fred Glover. A very large-scale neighborhood search algorithm for the multi-resource generalized assignment problem. *Discrete Optimization*, 1(1):87–98, 2004.

[34] Mutsunori Yagiura, Akira Komiya, Kenya Kojima, Koji Nonobe, Hiroshi Nagamochi, Toshihide Ibaraki, and Fred Glover. A path relinking approach for the multi-resource generalized quadratic assignment problem. In *Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics*, pages 121–135. Springer, 2007.

[35] Mutsunori Yagiura, Takashi Yamaguchi, and Toshihide Ibaraki. A variable depth search algorithm with branching search for the generalized assignment problem. *Optimization Methods and Software*, 10(2):419–441, 1998.

[36] Jirawan Yosnual and Srisawat Supsomboon. An integer programming for sugarcane factory supply allocation. In *Proceedings of the Fifth Asia Pacific Industrial Engineering and Management Systems Conference*, volume 21, pages 1–21, 2004.