Planificación dinámica de despachos de alimentos saludables en la ciudad de Santa Fe

Ignacio Vitale[1] and Rodolfo Dondo[2]

[1] Facultad de Ingeniería Química, U.N.L, Santiago del Estero 2829, 3000 Santa Fe, Argentina
vitalenacho@gmail.com
[2]Instituto de Desarrollo Tecnológico para la Industria Química (U.N.L. - Conicet), Güemes 3450, 3000 Santa Fe, Argentina
rdondo@santafe-conicet.gov.ar

Resumen. El problema de ruteo dinámico de vehículos (DVRP - *dynamic vehicle routing problem*) tiene como objetivo la determinación de un conjunto óptimo de rutas a fin de cumplir con las demandas de un conjunto de clientes en la medida en que los pedidos de dichos clientes arriban mientras se ejecutan las rutas. En la práctica, se presentan diferentes restricciones relacionadas a la satisfacción de los requerimientos del cliente y a limitaciones relacionadas con los transportistas y/o vehículos. Debido a la combinación de la planificación en tiempo real con la dificultad combinatoria presentada por el problema del diseño de rutas de distribución y dados los beneficios económicos potenciales asociados a una correcta planificación en línea, se ha dado especial atención al desarrollo de soluciones heurísticas con el fin de resolver ejemplos tomados de un caso real de una empresa dedicada a la comercialización y distribución de alimentos saludables en la Ciudad de Santa Fe.

1 Introducción

El problema de ruteo de vehículos consiste en la determinación del conjunto de rutas a seguir por cada vehículo a fin de satisfacer una serie de demandas. Los vehículos parten desde un centro de distribución; visitan a los clientes asignados y retornan al punto de partida para luego volver a cargar y distribuir, o para esperar el arribo de nuevas órdenes de servicio. En la práctica, estas demandas tienen asociados diferentes requerimientos de servicio impuestos por el cliente y cada vehículo utilizado para satisfacerlas cuenta con restricciones que limitan la calidad de servicios a prestar. En el presente trabajo se abordó una variación del problema de ruteo de vehículos con ventanas de tiempo y restricciones de capacidad heterogéneas en tiempo real. Entre sus complejidades cabe destacar que el problema se desarrolla en un entorno dinámico donde, la dimensión temporal desempeña un papel crucial en la toma de decisiones ya sea para definir, qué información va a ser procesada, cuándo será procesada y de qué manera va a ser procesada. En este ambiente, el objetivo del planificador difiere del objetivo planteado en un contexto estático (minimización de los costos de transporte) puesto que deben considerarse conceptos tales como prontitud de servicio y/o número de pedidos cumplimentados. Por lo tanto, la capacidad para re-direccionar en

adfa, p. 1, 2011.

tiempo real a un vehículo en ruta puede implicar ahorro de tiempos y costos. Además, ciertas demandas pueden o deben ser reprogramados para otro ciclo de entregas debido a la saturación de la capacidad de servicio. La problemática de distribución abordada tiene una particularidad que permite catalogar al problema como dinámico: la evolución de la información en el tiempo. Puesto que la información necesaria para la planificación de las rutas va siendo revelada al mismo tiempo que se efectúan las entregas, se deben tener en cuenta las siguientes cuestiones a la hora de obtener soluciones:

- *Decisiones de re-secuenciamiento y reasignaciones*: las soluciones iniciales generadas como óptimas pueden volverse sub-óptimas al revelarse nueva información sobre demandas a satisfacer.
- *Mecanismos de actualización de la información*: aunque se tenga acceso a toda la información, el mecanismo de actualización puede utilizar solo la información necesaria para generar soluciones en el corto plazo.
- *Tiempos de procesamiento para la obtención de soluciones*: las restricciones de tiempo de cómputo implican la construcción de soluciones en tiempos reducidos, ya que se desea obtener las rutas lo antes posible.
- *Metodología para la obtención de la solución*: es de suma importancia la utilización de algoritmos heurísticos y/o meta-heurísticos para la obtención de soluciones de buena calidad debido a la reducida disponibilidad de tiempo de procesamiento (en el orden de segundos).

En el presente trabajo se desarrollaron diferentes *mecanismos de actualización de la información* que se encargan de comunicar o revelar la información a los algoritmos utilizados como *metodología para la obtención de soluciones,* en base a determinados eventos que ocurren a lo largo del tiempo. Estas metodologías utilizadas para la obtención de soluciones fueron construidas con base en una adaptación de la metaheurística de Recocido Simulado (Simulated Annealing), la cual se encarga de procesar la información que proveen los mecanismos de actualización a fin de generar los recorridos a ser efectuados por cada vehículo con el objetivo de minimizar una función objetivo fijada por el usuario del procedimiento.

2 Descripción del problema

WINKA es una empresa que se dedica a la venta de alimentos y bebidas naturales a través de su sitio *e-commerce* y otros medios de comercialización. Como parte de sus servicios ofrece la distribución de sus productos mediante una flota de vehículos propia en el horario que va de las 8:00 a las 17:00 hs, debiendo cumplimentar los pedidos realizados por sus clientes con no menos de 2 horas de anticipación en la ciudad de Santa Fe y sus alrededores. Sus recursos de transporte tienen limitaciones de capacidad y difieren de un vehículo a otro. El volumen transportado por la flota puede variar tanto en la operación diaria, así como en los distintos días en los que brindan servicios de distribución. Esta parti-

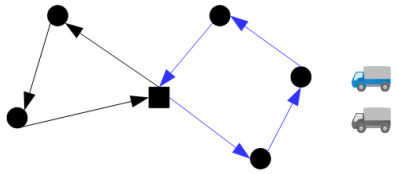


Fig. 1.a Contexto estático

cularidad es de suma importancia, ya que los recursos disponibles al momento de planear y re-planear las rutas puede llevar a diversas soluciones. Para visualizar las posibles decisiones a tomar se expone el siguiente ejemplo: supóngase que se tiene un conjunto pedidos a satisfacer mediante una flota de vehículos disponibles al momento de planificar las rutas. Usualmente, en un contexto estático, se minimizan los costos totales necesarios para visitar cada destino sin violar las restricciones planteadas. En el contexto de la problemática de distribución de WINKA, la disponibilidad de vehículos es de crucial importancia y por lo tanto un objetivo podría consistir en maximizar la cantidad de demandas satisfechas por unidad de tiempo utilizando la menor cantidad de vehículos posibles. Para poder llevar el mismo a cabo, es necesario un módulo de re-cómputo de rutas que debe ejecutarse periódicamente para conocer el estado de los vehículos en el corto plazo. Los posibles estados de un vehículo pueden ser: *ocioso, en ruta, o sirviendo*. Esto puede llevar a diferentes escenarios como los ilustrados en las figuras 1.b y 1.c.

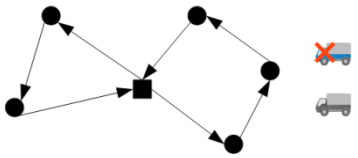


Fig. 1.b Se podrían generar múltiples rutas para un vehículo a fin de satisfacer las limitaciones de capacidad y ventanas de tiempo dejando al resto de la flota ociosa hasta la llegada de nuevos pedidos. Mientras más sean los vehículos disponibles, la empresa cuenta con mayor flexibilidad para responder a futuras demandas.

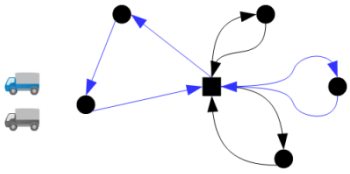


Fig. 1.c Otra alternativa puede ser la generación de múltiples rutas para cada vehículo en las que retornan al centro de distribución ya sea debido a ventanas de atención muy distantes entre los diferentes destinos o por consolidación de órdenes que superan las capacidades del vehículo asignado.

En WINKA se visualizó a la distribución como una ventaja competitiva frente a sus competidores ya que una eficiente distribución permite brindar mayor calidad de servicio a sus clientes; maximizar la cantidad de demandas satisfechas; minimizar los tiempos de atención entre pedidos; optimizar la utilización de la flota de vehículos disponibles; reducir costos, kilometrajes recorridos, tiempos de atención de pedidos, combustible utilizado y horas extras utilizadas en la atención de pedidos fuera del horario de trabajo habitual.

3 Estrategias de consolidación de órdenes

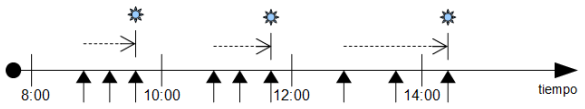
Previamente al procesamiento de la información necesaria para generar la planificación de las rutas, es esencial definir cuáles serán los “inputs” a ser entregados al algoritmo y el momento en el que se entregarán. La idea principal de estas políticas es postergar la ejecución de la planificación, hasta la ocurrencia de algún evento que libere las entradas para que sean comunicadas al algoritmo de ruteo [1]. Entre los posibles eventos que pueden desencadenar la re-planificación se destacan:

- *Eventos desencadenados por la recepción de nuevos pedidos.*
- *Eventos desencadenados por el cambio de estado de los vehículos disponibles*.

Para la problemática abordada se propusieron tres estrategias de consolidación de órdenes basadas en los estados de cada vehículo disponible, la evolución de las órdenes en el tiempo, la prontitud del servicio y el número de órdenes a satisfacer.

a) Consolidación de *m* ordenes de servicio: esta estrategia se encargará de actualizar las entradas del módulo de ruteo luego de que hayan arribado *m* órdenes de servicio a ser satisfechas por los vehículos disponibles.

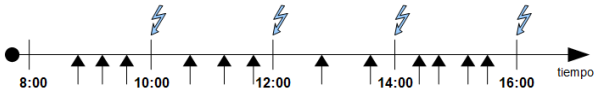
Fig. 2.a: Reactualización por número de ordenes arribadas.



En la figura (2.a) se pueden visualizar la ejecución del algoritmo luego de la recepción de $m = 3$ órdenes de pedido. Una de las desventajas de esta metodología es que ignora la evolución del tiempo, dejando de lado conceptos como la prontitud del servicio.

b) Re-planeación en intervalos fijos de tiempo: esta estrategia se encargará de entregar las entradas al módulo de ruteo luego de que haya transcurrido un intervalo de tiempo Δt fijo.

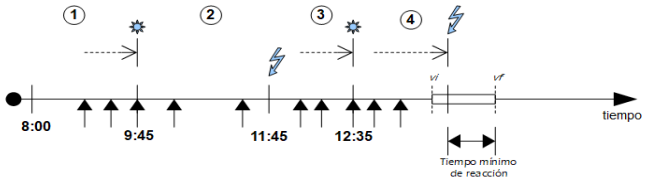
Fig. 2.b: Reactualización por intervalos de tiempo fijo.



En la figura (2.b) se puede visualizar la ejecución del algoritmo en intervalos Δt = 2hs. La desventaja de dicha metodología es que no tiene en cuenta el volumen de pedidos acumulados en los tiempos fijados, lo que puede saturar la capacidad de vehículos y generar soluciones no-factibles en base a los recursos disponibles al momento de re-planificar.

c) Estrategia mixta de re-planeación: Esta metodología consiste en un mix de las estrategias antes mencionadas y fue seleccionada para entregar información al algoritmo de diseño de rutas. En este caso, la comunicación de los inputs puede ser desencadenada ya sea por la llegada de *m* ordenes de servicio; por haber transcurrido Δt unidades de tiempo desde la última vez que fue ejecutado el algoritmo de ruteo o por haber alcanzado un valor tope definido como *tiempo mínimo de reacción*. El *tiempo mínimo de reacción* (*TR*) se define como la diferencia entre el *horario mínimo de atención* correspondiente a las órdenes en cola y el tiempo de viaje entre dicho destino y la base, sujeto a un cierto grado de incertidumbre. La metodología se ilustra mediante la siguiente figura:

Fig. 2.c: Reactualización mixta



En el caso que arriben $m = 3$ órdenes de servicio, el mecanismo comunicará los inputs. Además, los intervalos fijados previamente se actualizarían en Δt horas luego de haberse generado una solución o se actualizan al valor de tiempo más cercano al horario de arribo del vehículo a la base, sujeto a incertidumbre. Esto es visible en los ciclos 1 y 3 de entregas. En el caso que $m < 3$, se entregará la información en los intervalos fijados, siempre y cuando el valor de *TR* no sea alcanzado antes del próximo horario de re-actualización. Esto se puede visualizar en el ciclo 2 de entrega. En el caso contrario, donde $m < 3$ y el valor de *TR* se alcance antes del próximo intervalo de re-actualización, se comunicarán los inputs recolectados hasta al momento a fin de evitar que las soluciones se tornen no-factibles y se deban realizar comunicaciones con el cliente para aplazar los horarios de atención. Esto es visible en el ciclo de entregas 4.

Como consecuencia de los eventos estocásticos intrínsecos en los problemas dinámicos de distribución, estas alternativas de actualización deben contar con información del estado del vehículo con el fin de comunicar con mayor certeza la disponibilidad de un vehículo para comenzar a atender las demandas en cola. Es importante remarcar que previo a la ejecución del algoritmo de ruteo se debe definir el punto temporal desde el cual parte cada vehículo desde el centro de distribución. Este valor puede conocerse con certeza en el caso de que los vehículos estén en estado ocioso, o podría ser un valor estocástico dependiente del tiempo estimado de retorno a la base que está definido por los ciclos de entregas antes previstos para los vehículos en ruta.

4 Planificación de despachos

Posteriormente a la comunicación de la información por parte del mecanismo de actualización de la información, se debe contar con una metodología de solución para la problemática en base a los inputs recibidos. Para el presente trabajo, se utilizó un procedimiento catalogado como "meta-heurístico". Podríamos definir a estos procedimientos como *"estrategias de alto nivel que se encargan de guiar a procedimientos subordinados de menor nivel con el fin de explorar y explotar el espacio de búsqueda pertinente a problemas de optimización. Para llevar a cabo esto, se utilizan estrategias de aprendizaje que se encargan de estructurar la información de diversas maneras, a fin de encontrar soluciones cercanas a la óptimas de manera eficiente"* [2]. La elección de algoritmos de optimización meta-heurística se basa en los siguientes criterios:

- El objetivo de este tipo de metodologías es encontrar soluciones de calidad a un bajo costo computacional, recorriendo un espacio de búsqueda grande.
- Suelen contar con mecanismos destinados a evitar quedar atrapado en determinadas áreas del espacio de búsqueda escapando de *mínimos/máximos locales*.
- La implementación de estas herramientas es relativamente sencilla y fácilmente adaptable a una gran variedad de aplicaciones prácticas.

¿Por qué no utilizar otras metodologías?

- El esfuerzo computacional consumido por **metodologías exactas** para alcanzar el óptimo global crece exponencialmente con el tamaño del problema. Esto no suministra garantía de que se obtendrá el óptimo en un tiempo de cómputo reducido.

- Las **heurísticas** siguen procedimientos específicos que solo se adaptan a determinadas situaciones y problemas. Estos procedimientos no brindan garantía de que se encuentren soluciones de buena calidad, cercanas al óptimo global, para situaciones alternativas a las que están destinadas.

4.1 Introducción a la metodología utilizada

En el presente trabajo se presenta una variación de la metodología de *Recocido Simulado* [3]. El autor se inspiró en el trabajo de *Metropolis* [4] y está basado en la siguiente analogía: Se parte de conjunto de átomos en equilibrio, a una temperatura específica. Luego, mediante simulación, un átomo es perturbado y el resultado del cambio en la energía del sistema es computado (ΔE). Si $\Delta E \leq 0$, la perturbación es aceptada y la configuración alterada se convierte en el punto de partida para la próxima iteración. Si $\Delta E > 0$, la probabilidad de que la nueva configuración sea aceptada está dada por la probabilidad de Boltzmann, $P(\Delta E) = \exp(-\Delta E/k_B T)$, donde T especifica la temperatura del sistema y k_B denota la constante de Boltzmann. Este valor es comparado con un número generado aleatoriamente siguiendo una distribución uniforme en el intervalo (0,1). Si es es menor que $P(\Delta E)$, la nueva configuración es aceptada; en el caso contrario, se rechaza. Kirkpatrick [3], inspirado en esta metodología, desarrolló un algoritmo en el cual la energía del sistema se reemplaza por una *función de costo* destinada a valorar la calidad de una solución. De esta manera implementó un procedimiento con capacidad para explorar el espacio de búsqueda de un problema combinatorio de manera simple y sencilla. El algoritmo de Recocido Simulado simula el proceso de recocido de sólidos a fin de resolver problemas de optimización. El mismo consiste en "calentar" una configuración del sistema para luego "enfriarla" en una serie de etapas sucesivas. En cada una, la temperatura disminuye lentamente, hasta "congelar" el sistema y alcanzar un estado en el cual no es posible introducir nuevos cambios. Al alcanzar altas temperaturas, los átomos aumentan su energía y pueden desplazarse desde sus posiciones iniciales. En los pasos subsiguientes, el sistema se "enfría" lentamente para incrementar la probabilidad de recristalizar en configuraciones de menor energía y, en consecuencia, de aumentar las probabilidades de alcanzar un *mínimo global de energía*.

El procedimiento se ilustra mediante la analogía representada en la Figura 3. Cada paso del procedimiento de "alto nivel" puede simbolizarse por un **peldaño**. El ***alto*** de cada uno de ellos representa la variación de la temperatura (ΔT) en cada iteración respecto de la anterior. El ***largo*** de cada escalón se asocia a un procedimiento de menor nivel definido como etapa de *búsqueda de vecindades*. Este mecanismo se encarga de generar nuevos arreglos, reorganizando la configuración actual de la solución. Una vez obtenida la nueva configuración, se valora su costo frente al último alma-

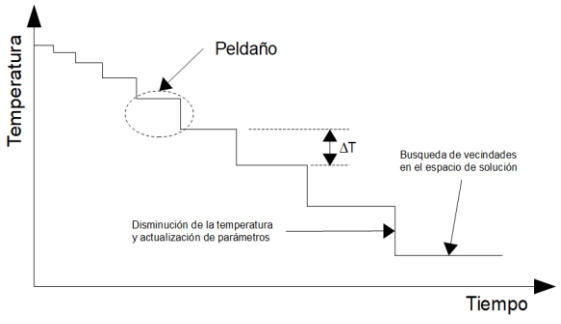


Fig. 3. Disminución de la temperatura en el tiempo

cenado. Si el valor obtenido para el nuevo arreglo es menor que el actual, es aceptado. Caso contrario, el nuevo arreglo podría ser aceptado en base a un criterio probabilístico. Este proceso se repite hasta alcanzar el final del **escalón**. En este punto se enfría el sistema con el fin de disminuir la probabilidad de aceptar soluciones con mayores costos. Este proceso de "alto nivel" se repite hasta alcanzar un criterio de terminación definido por el usuario.

4.2 Adaptación a la problemática de WINKA

La metodología se inicia determinando la cantidad de recorridos que puede realizar cada vehículo en base a los vehículos disponibles, su capacidad y las demandas efectuadas por los clientes. En esta etapa, se define la cantidad de recorridos posibles por cada vehículo como el número de demandas a satisfacer. Una vez definidas las entradas, se debe disponer de una variable solución que sea fácil de alterar e interpretar por el algoritmo. Para el presente trabajo se utilizó una codificación vectorial en forma de cromosoma. Los números enteros indican los índices de los pedidos a ser atendidos y las letras "v" indican el recorrido en orden de ocurrencia para cada vehículo. Así, todos los enteros que se encuentren antes de cada "v" indican, de izquierda a derecha, el orden de visita a efectuarse por cada vehículo en cada uno de sus recorridos. Esto se ilustra en el siguiente ejemplo: la empresa cuenta con dos vehículos que pueden realizar dos recorridos para satisfacer nueve demandas. Una solución se representa de la siguiente forma:

S = [v, 8 , 5 , 3 , v , 2 , 1 , v , 4 , 6 , 7 , v , 9]

Vehículo 1: { *Recorrido 1*: [Base, 8 , 5 , 3 , Base] ; *Recorrido 2*: [Base, 2 , 1 , Base] }
Vehículo 2: { *Recorrido 1*: [Base, 4 , 6 , 7 , Base] ; *Recorrido 2*: [Base, 9 , Base] }

4.2.1 Parámetros del algoritmo

Los parámetros utilizados por el algoritmo son los siguientes:

Fijos:

- α: Factor utilizado para alterar el número de iteraciones
- Ø: Penalización por sobre-capacidad del vehículo
- $H_{máx}$: Tiempo límite de ejecución del algoritmo
- T_0: Temperatura inicial del sistema
- μ: Penalización por el número de vueltas efectuadas

Variables:

- β: Factor de enfriamiento utilizado para disminuir la temperatura del sistema
- λ_{tz}: Penalización por ventanas de tiempo violadas
- T: Temperatura del sistema
- L: Largo del peldaño/Número de iteraciones ejecutadas a T constante

4.2.2 Diagrama de flujo del procedimiento (Figura 4)

Habiendo especificado el número de posibles recorridos, se genera una solución aleatoria que representa el estado inicial del sistema. A partir de esta, se evalúa su costo y se almacena como la mejor solución encontrada hasta el momento.

Se inicializa el procedimiento con la solución inicial y se define: el número de iteraciones *L* que deben ejecutarse para descender al próximo escalón, la temperatura inicial del sistema T_0, los parámetros de penalización de costos (λ_{tz}, μ, Ø) y los factores que alteran los parámetros del modelo (α, β). Luego se comienza a recorrer cada **peldaño** hasta alcanzar un tiempo límite $H_{máx}$ (criterio de terminación). Cada escalón se recorre de la siguiente manera hasta alcanzar el número *L* que especifica el largo del mismo.

En una primera instancia se perturba la última solución aceptada mediante uno de los mecanismos de perturbación utilizados con el fin de generar una nueva configuración. Luego se evalúa para conocer su calidad y, según el mecanismo definido por Metropolis, si el costo de la nueva configuración es menor que el actual, se acepta y almacena como punto de partida para la próxima iteración. Paso siguiente, se compara dicha solución con la mejor encontrada hasta el momento. Si este es el caso, la solución se almacena como la mejor encontrada hasta el momento; de no ser así, el procedimiento continúa. Siguiendo el flujo alternativo, se utiliza un criterio de aceptación dependiente de la temperatura del sistema y, la diferencia entre el costo del nuevo arreglo y el correspondiente a la última configuración encontrada. En este caso, si la solución se acepta, se computa su costo y se utiliza como punto de partida para la próxima iteración, caso contrario, el flujo continúa.

Una vez alcanzado el número de iteraciones *L*, se actualiza la temperatura del sistema, el largo del escalón y los parámetros de penalización utilizados. Luego, se recalculan el costo de la última solución almacenada y el de la mejor solución encontrada a fin de medirlos con los nuevos parámetros de penalización. Finalizada la actualización, se desciende al próximo **peldaño** para repetir el procedimiento hasta alcanzar el criterio de terminación definido por $H_{máx}$.

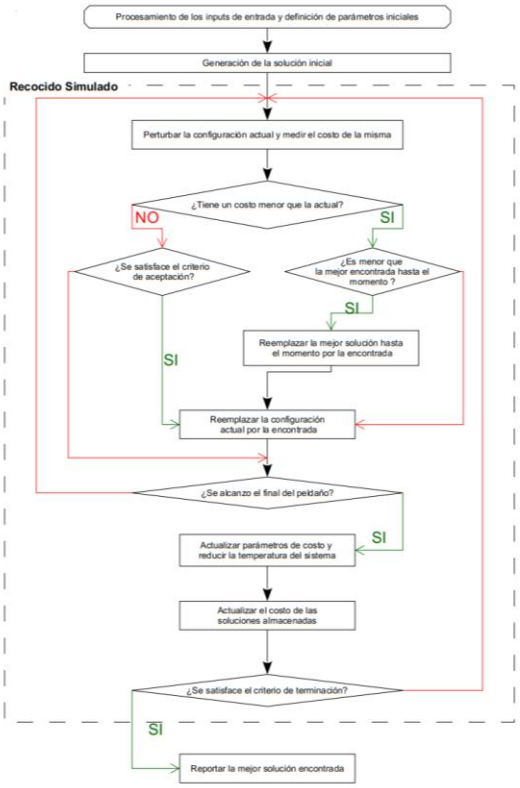


Fig. 4. Metodología de asignación de órdenes

4.2.3 Criterio de aceptación

Este criterio se encarga de decidir si una nueva solución será aceptada. Una nueva configuración reemplaza a la actual siempre y cuando su costo sea menor al costo de la última configuración almacenada. En el caso contrario, la nueva solución puede reemplazar a la solución almacenada con una probabilidad definida por las ecuaciones (1) y (2):

$$P\left(\text{Costo}_{(\text{nueva})}, \text{Costo}_{(\text{actual})}, \text{T}\right) = e^{\frac{-\Delta Costo}{T}} \quad (1)$$

$$\Delta Costo = \left|\text{Costo}_{(\text{nueva})} - \text{Costo}_{(\text{actual})}\right| \quad (2)$$

Donde $\text{Costo}_{(\text{nueva})}$ es el costo de la nueva solución; $\text{Costo}_{(\text{actual})}$ es el costo de la solución antigua; T es la temperatura específica del sistema y $P\left(\text{Costo}_{(\text{nueva})}, \text{Costo}_{(\text{actual})}, \text{T}\right)$ es la probabilidad de aceptar a la solución antigua por la nueva. Sea *Rand* un número aleatorio generado según una distribución uniforme en el intervalo [0,1]. Si *Rand* < $P(C_n;C_a;T)$ la nueva solución reemplaza a la antigua y en el caso contrario, se rechaza.

4.2.4 Actualización de parámetros

Para evitar soluciones no factibles (es decir incumplimiento de los requerimientos de cada cliente) es necesario actualizar los parámetros de penalización utilizados para explorar el espacio de solución con el fin de evitar quedar atrapado en *mínimos locales*. Así, el algoritmo realiza esta tarea comenzando con un valor por penalización de vueltas (μ) elevado y un valor del parámetro de violación de ventanas de tiempo (λ_{tz}) muy pequeño a fin de explorar soluciones que aprovechen cada vehículo al máximo posible, pero, con altas probabilidades de violar los horarios de atención definidos. Al finalizar cada **peldaño**, λ_{tz} se incrementa gradualmente con las violaciones de las restricciones impuestas, según las ecuaciones (3), (4) y (5):

$$\lambda_{tz} = \lambda_{tz} + \min(w, z) * Reloj \quad (3)$$

donde:

$$w = \textit{Porcentaje de veces que se violan las ventanas de tiempo} = \frac{\#\ de\ ordenes\ atendidas\ fuera\ del\ horario\ pactado}{\#\ de\ ordenes\ atendidas} \quad (4)$$

$$z = \textit{Sumatoria de los incumplimientos de horarios de atención para cada orden p} = \frac{\sum_p \max(0, Horario\ entrega_p - Tiempo\ máximo\ de\ atencion_p)}{Tiempo\ en\ ruta} \quad (5)$$

A medida que transcurre el tiempo de ejecución, la penalización λ_{tz} crecerá con el objetivo de fortalecer las restricciones de ventanas de tiempo impuestas, en la medida que se violen. Este incremento elevará la cantidad de vehículos requeridos para satisfacer las órdenes ya que el parámetro de penalización por vueltas utilizadas μ se mantiene fijo.

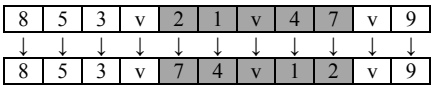
El algoritmo se inicializa con valores de L pequeños que se van incrementando gradualmente. En el caso de T, este parte de un valor máximo T_0 y disminuye gradualmente con la ejecución de cada peldaño como se puede visualizar en la figura 4.1. Una vez que la temperatura alcanza un valor mínimo ($T_{mín}$), esta se reactualiza a su valor inicial T_0.

La penalización Ø utilizada para evitar sobrepasar el límite de capacidad disponible en cada vehículo se establece con valores extremadamente grandes para evitar soluciones que violen esta restricción.

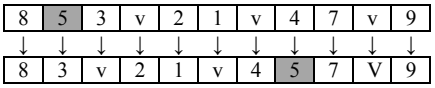
4.2.5 Métodos de permutación utilizados

Se utilizaron tres tipos de permutaciones para alterar la configuración del sistema [5]:

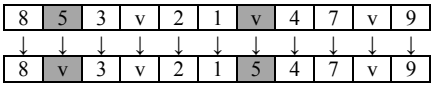
Mutación por inversión: Se seleccionan dos posiciones al azar dentro de la configuración y se invierten el(los) sub-tour(s) de la solución.



Mutación por inserción: Se selecciona una posición al azar e inserta en otra posición aleatoria de la configuración sin alterar al resto del arreglo.



Intercambio 2-OPT: Se seleccionan dos posiciones al azar y se invierten las posiciones en la configuración.



4.3 Pseudocódigo del procedimiento de Simulated Annealing

```
Inicio
Recepción de datos
Cálculo de los parámetros iniciales
Definir α, Ø, H_máx, T_0, β, λ_tz, μ, T, L
Generar solución inicial S(actual)
Fijar Reloj = 0
Fijar S(best) = S(actual) ; CT(best) = CT(actual)
  While (Reloj < H_máx)
    For j = 1 to L
      Generar una nueva solución S(nueva)
       If CT(nueva) <= CT(actual)
          Fijar S(actual) = S(nueva)
                CT(actual) = CT(nueva)
       Otherwise
         Generar PB = P(CT(nueva),CT(actual),T)
                Pr = Aleatorio([0,1])
         If PB > Pr
           Fijar S(actual) = S(nueva)
                 CT(actual) = CT(nueva)
         End if
       End if
       If CT(nueva) < CT(best)
          Fijar S(best) = S(nueva)
                CT(best) = CT(nueva)
       End if
    Next j
    Actualizar T ; L ; β ; λ_tz ; Reloj
    Actualizar CT(best) y CT(actual)
  End while
  Reportar S(best)
Fin
```

Se definen todos los parámetros y variables a utilizar por el algoritmo.

Se inicializa con una solución inicial generada aleatoriamente

Se define la mejor solución encontrada como la inicial

Se utilizan los métodos de permutación para alterar la solución actual y generar una nueva

Se compara el costo de la nueva solución frente al actual

Se calcula la probabilidad de aceptar la solución y se la compara con un número generado aleatoriamente en el intervalo [0,1]

Se compara el costo de la nueva solución frente a la mejor encontrada hasta el momento

-Se actualizan todos los parámetros de costo y el Reloj del sistema
-En base a los nuevos parámetros de costo se actualizan los costos de las soluciones almacenadas
-Se reporta la mejor solución almacenada hasta el momento

5 Simulación y resultados

A fin de evaluar la metodología desarrollada se realizaron simulaciones con el objetivo de demostrar su potencialidad para alcanzar soluciones de buena calidad en tiempos de ejecución reducidos. Para ello se codificó el algoritmo en Python 3.7, y todas las pruebas se efectuaron en una Notebook con un procesador Intel Core i7-4510U, 2.6 GHz y 8 GB de RAM.

5.1 Descripción de la simulación

Para imitar la operación diaria de WINKA, el simulador, ilustrado en la figura 5, debe chequear una vez por minuto si han arribado nuevas órdenes y el estado de los vehículos en operación. En el caso que ocurra algún evento que concluya en la ejecución del algoritmo de solución, el mecanismo de actualización de la información se encargará de transmitir la información necesaria para generar las asignaciones de las demandas a cada vehículo en operación [6].

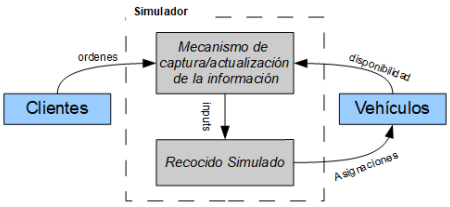


Fig. 5. Componentes del Simulador

5.2 Problema simulado

La problemática se ilustra mediante la resolución de un ejemplo artificial creado en base a información de órdenes reales de la empresa y suponiendo el comienzo de la operación a partir de las 8:00 hs. En el mismo se consideran veinticinco órdenes generadas a lo largo de un día de trabajo entre las 8:00 y las 16:00 hs.

Cada cliente cuenta con un tiempo de servicio in-situ de 10 minutos y, cada demanda fue generada según una distribución uniforme en el intervalo [15;65] unidades.

Para simular el problema, se dividió el horizonte de tiempo en cinco horizontes de dos horas. Teniendo en cuenta cada etapa se estimó el número de órdenes atendidas en cada una de ellas. A partir de cada intervalo, se definió la ventana de tiempo inferior como el horario inicial de cada etapa más un número aleatorio multiplicado por 120 minutos. La ventana de tiempo superior de cada orden se definió con una holgura de 2 horas partiendo de la ventana inicial. Por último, se definió al tiempo de arribo de la orden como la ventana de tiempo inferior menos un número aleatorio, dentro del intervalo [0, 1], multiplicado por 120 minutos. Las órdenes recibidas antes de las 8:00 hs. están disponibles al comienzo de la planificación y solo se aceptan órdenes hasta las 16:00 hs. Por lo tanto, toda orden recibida posteriormente deberá ser atendida en el día siguiente y estará disponible al comienzo del próximo período de planificación. La operación de distribución fue simulada con la disponibilidad de dos vehículos, con capacidad de 100 unidades, partir de las 8:00 hs.

5.3 Resultados obtenidos

La simulación del problema fue ejecutada con los siguientes parámetros:

- **Número de órdenes en cola**: $m = 8$.
- **Intervalos de reactualización a tiempo fijo**: 1 hora.
- **Criterio de terminación**: 10 segundos de ejecución

Los resultados obtenidos pueden visualizarse en la Tabla 1:

Tabla 1. Asignaciones realizadas en los correspondientes re-cómputos

Re-cómputo	Pedidos	V1		V2	
		Hora entrega	**Recorrido**	**Hora entrega**	**Recorrido**
1	1	08:12	1	**Disponible**	
2	2	**No disponible**		08:30	1
3	3	**Disponible**		10:11	2
4	6	10:25	2	**No disponible**	
	5 4	11:08 11:25	3		
5	7	**No disponible**		11:19	3
6	9 8	12:35 12:53	4	**No disponible**	
7	10	**No disponible**		13:09	4
8	14 11 13	14:01 14:13 14:26	5	**Asignado**	
	17 18	**Asignado**		14:19 14:31	5
	15 12			14:53 15:10	6
9	19 16 20	14:55 15:16 15:32	6	**No disponible**	
10	21 22 23	**No disponible**		16:09 16:33 16:55	7
11	24 25	17:20 17:37	7	**No disponible**	

En la tabla 1 pueden visualizarse cada re-cómputo como la ocurrencia de algún evento que desencadena la comunicación de los inputs a la meta-heurística para que asigne las órdenes pendientes a los vehículos disponibles. En cada re-cómputo se puede contar con uno o más vehículos según su disponibilidad a la hora de comunicar

las entradas al algoritmo. Esto se visualiza mediante los estados *Disponible* y *No disponible* correspondientes a cada vehículo a la hora de planificar los despachos. En el caso de que en un mismo re-cómputo se asigne más de un vehículo para satisfacer las órdenes pendientes, se indica con el estado *Asignado* que el vehículo fue utilizado para atender otras órdenes en el mismo período. Es importante aclarar que cada vehículo está *Disponible* siempre y cuando el horario de arribo al punto de partida sea menor o igual al momento en el cual se lleva a cabo la planificación de los despachos. Caso contrario, dicho transportista se encuentra en el estado *No disponible*. Los recorridos de cada vehículo (*V1* y *V2*) se componen de una serie de pedidos ordenados según la secuencia en la que son visitados, culminando la ruta en el punto de partida. Para cada pedido se especifica el horario de finalización del servicio mediante la columna denominada "Hora entrega".

Es importante remarcar que, el criterio utilizado para definir la disponibilidad de los vehículos a la hora de atender las órdenes no utiliza las estimaciones de tiempo sobre los horarios de arribo de los vehículos en ruta; esta variación no fue ejecutada en la simulación y, por lo tanto, no se permitió la utilización de múltiples vehículos en futuros despachos según estimaciones de los horarios de arribo.

6 Conclusiones

En el presente trabajo se abordó un problema de ruteo de vehículos en un entorno dinámico que surge a partir de la necesidad de mejorar la eficiencia de la operación logística en una empresa santafesina dedicada a la comercialización y despacho de productos alimenticios. Para dar solución a la problemática se desarrollaron estrategias de procesamiento de la información sobre las órdenes a satisfacer. Luego se definió un algoritmo de ruteo encargado de asignar las órdenes en cola a los vehículos disponibles. Se utilizó la metodología de Recocido Simulado para generar asignaciones de órdenes a vehículos. También se utilizó la información de salida del algoritmo para transferir la información utilizada con el fin de simular los recorridos y los horarios en los cuales los vehículos vuelven a estar disponibles para satisfacer nuevas órdenes.

Estudios computacionales adicionales no reportados en este trabajo fueron llevados a cabo sobre una serie de problemas estáticos con el fin de demostrar que el algoritmo propuesto tiene la capacidad de alcanzar resultados de buena calidad en condiciones de tiempos de cómputo limitados. Esto nos lleva a concluir que la metodología desarrollada posee la capacidad para resolver problemas de tamaño mayor al presentado con propósitos ilustrativos. Sin embargo, se destaca que la performance se encuentra estrechamente vinculada a la evolución de sus parámetros en el tiempo. Por lo tanto, resulta imprescindible analizar la información que es ingresada al algoritmo de ruteo con el fin de adaptar los parámetros iniciales y la forma en la que evolucionan si se desea un mejor desempeño del mismo.

Actualmente WINKA se encuentra en etapa de implementación de la metodología presentada en este trabajo. Posteriormente a esta etapa se deberán ajustar los parámetros del algoritmo a situaciones particulares de la empresa con el fin de mejorar la performance del mismo.

7 Referencias

1. Larsen, A. 2001. The Dynamic Vehicle Routing Problem. PhD thesis, Technical University of Denmark (DTU).
2. Osman, I. H. and Laporte, G. 1996. Metaheuristics: A bibliography. Annals of Operations Research. 63, 513–623.
3. Kirkpatrick, S., Gelatt, Jr. C.D., and Vecchi, M.P. 1983. Optimization by simulated annealing, Science, 220, 671–680.
4. Metropolis, N.; Rosenbluth, A.W.; Rosenbluth, M.N.; Teller, A.H.; Teller, E. 1953. Equation of State Calculations by Fast Computing Machines. Journal of Chemical Physics. 21 (6): 1087–1092.
5. Schneider, J.J, and Kirkpatrick, S. 2006. Stochastic Optimization. Sringer.
6. Montemanni, R., Gambardella, L.M., and Rizzoli, A.E. 2005. Ant Colony System for a Dynamic Vehicle Routing Problem. Journal of Combinatorial Optimization 10: 327.