Planeación reactiva de plantas “batch” multiproducto, multietapa basada en programación con restricciones

Franco M. Novara#*, Gabriela P. Henning*

#Facultad de Ingeniería Química, UNL, Santiago del Estero 2829, 3000 Santa Fe, Argentina
*INTEC (UNL, CONICET), Güemes 3450, 3000 Santa Fe, Argentina

fra_novara@hotmail.com,ghenning@intec.unl.edu.ar

Abstract. Se presenta una metodología sistemática para abordar el problema de “scheduling” reactivo, es decir, la planeación a corto plazo de una planta “batch” multi-producto, multi-etapa ante la ocurrencia de eventos imprevistos, tales como la salida de funcionamiento de un equipo o la llegada de una nueva orden. El enfoque propuesto apunta a preservar la estabilidad de la agenda original, minimizando el número de cambios que sobre ella se realizan, a la vez que se trata de mantener un buen desempeño con relación a la medida de performance con la que se desarrolló el “schedule” vigente, que generalmente es “Makespan”. Para atender a estos requerimientos se desarrolla un enfoque multi-objetivo, el cual se valida a través de la solución de múltiples ejemplos.

Keywords: “Scheduling” reactivo, Programación con restricciones, Plantas “batch” multi-producto, multi-etapa, Estabilidad de la agenda de trabajo

1 Introducción

Se aborda el problema de programación de la producción reactivo (“scheduling reactivo”) de una planta “batch” multiproducto, multietapa, cuyo objetivo es corregir o modificar una agenda de trabajo que se ve afectada por eventos tales como la falla de una unidad de procesamiento o la llegada de una nueva orden de producción. A pesar de su gran interés práctico, recién en la última década fue considerado con mayor fuerza por la comunidad académica [1]. El presente trabajo se focaliza en esta problemática, mediante el desarrollo de una metodología basada en programación con restricciones (“Constraint Programming”, CP). La misma emplea un novedoso enfoque multi-objetivo que apunta a preservar la estabilidad de la agenda original, minimizando el número de cambios que sobre ella se efectúan, a la vez que se trata de lograr un buen desempeño con relación a la medida de performance con la que se generó el “schedule” vigente, que generalmente es “Makespan”.

En la Sección 2 de este trabajo se describe el problema abordado, así como las suposiciones realizadas. En la siguiente sección, se presenta la metodología y el modelo propuesto. En la Sección 4 se discuten resultados computacionales. Finalmente, se presentan conclusiones y trabajos futuros.

2 Descripción del problema

Se considera un ambiente productivo "batch" en el que en cada etapa se dispone de equipos operando en paralelo, los que pueden tener características diferentes (tiempos de procesamiento distintos para un mismo "batch"). Puede ocurrir que no todas las unidades de una etapa sean aptas para la fabricación de un determinado "batch". Asimismo, pueden existir secuencias de procesamiento (productos fabricados uno a continuación del otro) prohibidas. Además, cuando se procesan dos "batches" consecutivos en un dado equipo, generalmente se requiere realizar tareas de limpieza ("changeover"), cuya duración depende de la secuencia de productos involucrados.

En este ambiente es necesario agendar, en cada período, un conjunto de órdenes monoproducto. Cada orden da origen a una serie de "batches" que deben atravesar secuencialmente las etapas del proceso. Otro aspecto a contemplar son las políticas de almacenamiento intermedio y de espera entre etapas, las cuales dependen de las recetas de los productos a fabricar y de las facilidades de almacenamiento existentes. En este trabajo se contempla trabajar (i) sin restricciones de almacenamiento intermedio, suponiendo que existe capacidad ilimitada (UIS, "Unlimited intermediate storage"), así como la situación opuesta, (ii) considerar que no existen tanques de almacenamiento intermedio (NIS, "nonintermediate storage"). En cuanto a las políticas de espera entre etapas, (i) se permitirá que un "batch" aguarde en el equipo que lo procesó, hasta que la siguiente unidad esté disponible, el tiempo que sea necesario (UW, "unlimited wait") dando lugar a una política NIS/UW; (ii) permitiendo esperas de tiempo limitado (FW, "finite-wait"), dando lugar a una política NIS/FW, o (iii) no admitiendo esperas (ZW, "zero-wait"), dando lugar a una política NIS/ZW.

La agenda de trabajo inicial ("schedule" predictivo) puede verse afectada por un evento inesperado, como ser el arribo de una nueva orden o la falla de una unidad. En este último caso se considerará que el "batch" que estaba siendo procesado se deteriora y deberá ser re-procesado. A partir de la ocurrencia del evento (cuyos datos se transmiten de forma instantánea), se relevará la información correspondiente al estado actual de la planta y al evento que ha ocurrido, y se los combinará con los datos del "schedule" vigente. Con toda esta información deberá modificarse la agenda de trabajo de forma tal de realizar la menor cantidad de cambios posibles en la misma, y a la vez obtener un "schedule" eficiente desde el punto de vista de la métrica utilizada para medir el desempeño de la agenda inicial, como por ejemplo, "Makespan". Para lograr un balance entre estos objetivos antagónicos, y a diferencia de la propuesta reportada en [2], se empleará un enfoque multi-objetivo en el cual las funciones de desempeño serán normalizadas para ponerlas en un pie de igualdad. La propuesta reportada en este trabajo posee claras ventajas sobre las reparaciones locales que, por lo general, realizan manualmente los planificadores en la práctica industrial.

Las modificaciones que se hagan al "schedule" vigente no tendrán el mismo impacto si los cambios se llevan a cabo en tareas próximas al tiempo en el que ocurre el evento imprevisto (tareas cuyo inicio es inminente) y que por ende no son deseables, que si las alteraciones afectan a tareas ubicadas al final de la agenda. Para contemplar este hecho se ha elaborado una clasificación conceptual de los intervalos temporales

asociados al "schedule", la cual extiende las nociones previamente desarrolladas en el grupo de trabajo y que fueran presentadas en [2].

La Figura 1 muestra los períodos en los que puede dividirse la agenda de trabajo actual. El intervalo denominado *"rescheduling time window"*, que se extiende desde el momento en que se produce el evento disruptivo ("event time") hasta que se implementa la agenda modificada en el *"implementation time point"*, es aquél durante el cual se llevará a cabo la actividad de "rescheduling" o revisión de la agenda por parte del planificador. También se define el denominado *"intervalo de freezing"*, cuya duración es función de la etapa de procesamiento y para cada una de ellas termina en un instante denominado "*freezing period end point*". En este intervalo no es posible realizar cambios en la agenda de las tareas que tenían su inicio previsto durante el mismo. Sólo se permitirán modificaciones en las actividades que correspondan a "batches" directamente afectados por la falla de la unidad (aquéllos que tenían una tarea asignada al equipo durante el período de indisponibilidad del mismo). Asimismo, se define un "*intervalo crítico"*, cuya extensión también es función de la etapa de procesamiento y que finaliza en un instante denominado "*critical period end point*", que es función de la etapa. Aquellas tareas que tenían inicio previsto durante este intervalo podrán ser alteradas (adelantamientos y atrasos en el tiempo de inicio, así como cambios en la unidad asignada), aunque estos cambios serán penalizados.

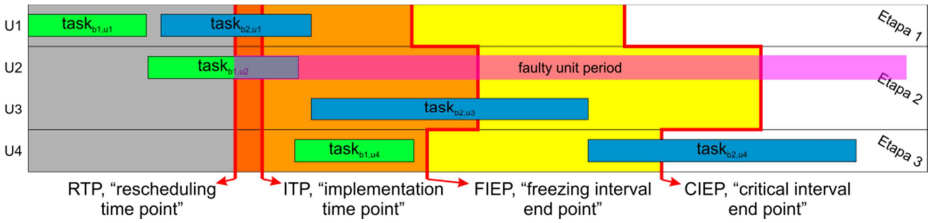

Fig. 1. Intervalos temporales en los que se divide la agenda de trabajo vigente

3 Metodología de "rescheduling" basada en un modelo CP

3.1 Nomenclatura

Conjuntos/Índices

B/b: "batches" requeridos en el horizonte de planeación.
P/p, p': productos demandados en el horizonte de planeación.
S/s: etapas de procesamiento.
U/u: unidades de procesamiento.
Fp: secuencias de procesamiento prohibidas representadas mediante pares, $f=<p,p'>$.

Parámetros

abP: "aborted batch penalty", penalidad relacionada a la interrupción definitiva de un "batch".
apt_s: "average processing time", tiempo promedio de procesamiento en la etapa *s*.
asP1, asP2, asP3: "anticipated start penalties", penalidades asociadas al adelanto del tiempo de inicio de procesamiento de un "batch" en una etapa.

bMk: “best Makespan”, “Makespan” de la agenda original.
$bpt_{p,u}$: “batch processing time”, tiempo de procesamiento de un “batch” de producto *p* en la unidad *u*.
cc: “critical coefficient”, factor de ponderación asociado al fin del intervalo $ciep_s$.
$ciep_s$: “critical interval end point”, finalización del período de criticidad en la etapa *s*.
cot_u: “changeover times”, matriz que contiene los tiempos de “changeover” en la unidad *u* para pares de productos que se procesan consecutivamente.
cuP1, cuP2, cuP3: “changed unit penalties”, penalidades ligadas al cambio de unidad en el procesamiento de un “batch” en una etapa.
dsP1, dsP2, dsP3: “delayed start penalties”, penalidades asociadas al retraso en el tiempo de inicio de procesamiento de un “batch” en una etapa.
et: “event time”, instante en el que falla una unidad o arriba una nueva orden.
fc: “freezing coefficient”, factor de peso asociado al fin del intervalo $fiep_s$.
$fiep_s$: “freezing interval end point”, tiempo de fin del intervalo de “freezing” en la etapa *s*.
itp: “implementation time point”, momento de implementación de la nueva agenda.
$mwt_{p,u}$: “maximum wait time”, tiempo de espera máximo de un “batch” de producto *p* en la unidad *u*, luego de finalizado su procesamiento.
pb_b: “planned batch”, “batch” perteneciente al “schedule” original, para el cual se especificará su “status” y su situación frente a su eventual revisión o “rescheduling” (atributos “status” y “sched”, respectivamente).
$pt_{b,u}$: “planned task”, tarea perteneciente a la agenda original, ligada a la ejecución del “batch” *b* en la unidad *u*, para la cual se conoce su tiempo de inicio y finalización.
rtp: “rescheduling time point”, tiempo de inicio del “rescheduling” e igual a *et.*
rw: “rescheduling window”, período comprendido entre el *rtp* y el *itp.*
u_f: “faulty unit”, unidad en la que se produce la falla.
urtp: “unit recovery time point”, momento en el cual u_f entra nuevamente en servicio.
wMk: “worst Makespan”, “Makespan” resultante de reagendar al final del horizonte de planeación todas las tareas asociadas a los “batches” interrumpidos que deben relanzarse, así como los que podrían reagendarse. Las tareas se incorporan a la agenda conservando la unidad original y, en caso de estar asignadas a un mismo equipo, manteniendo el orden relativo entre sí.
wP: “worst penalty”, penalidad más alta que podría esperarse.

Variables

ab_b: “aborted batch”, variable binaria que identifica cuando el “batch” *b* es interrumpido en forma definitiva.
abTP: “aborted batch total penalty”, penalidad relacionada a los “batches” abortados.
$as_{b,s}$: “anticipated start”, anticipación con la que el “batch” *b* comienza a ejecutarse en la etapa *s*, respecto al tiempo planificado en la agenda original.
$asb_{b,s}$: variable binaria asociada con un inicio anticipado del “batch” *b* en la etapa *s*.
$cu_{b,u}$: “changed unit”, variable binaria que toma el valor 1 cuando, al realizarse el “rescheduling”, el “batch” *b* asignado a la unidad *u* cambia de equipo en dicha etapa.
cuTP: “changed unit total penalty”, penalidad relacionada a cambios de equipos de los distintos “batches” respecto a lo inicialmente planeado.

$ds_{b,s}$: "delayed start", retraso con el que el "batch" *b* comienza a ejecutarse en la etapa *s*, respecto al tiempo previsto en el "schedule" inicial.
$dsb_{b,s}$: variable binaria asociada con un inicio atrasado del "batch" *b* en la etapa *s*.
$task_{b,u}$: variable de intervalo que representa la ejecución del "batch" *b* en la unidad *u*.
$stTask_{b,s}$: variable de intervalo que representa la ejecución del "batch" *b* en la etapa *s*.
$unitBatchSeq_u$: variable de secuencia definida para cada unidad *u*. Representa un ordenamiento de las variables de intervalo $task_{b,u}$ asociadas a la unidad *u*. Cada tarea está caracterizada por un tipo que es igual al producto *p* que se fabrica en el "batch" *b*.
tcTP: "time change total penalty", penalidad relacionada al cambio en los tiempos de inicio de las tareas.

3.2 Clasificación de las tareas y "batches" de la agenda original

Clasificación de las tareas que componen la agenda de trabajo

Las expresiones lógicas (1)-(5) permiten clasificar las tareas que componen la agenda inicial, asignándoles un estado ("status"). Las expresiones (1) y (2) identifican las tareas que han sido canceladas; la (3), a aquéllas que han finalizado o finalizarán su ejecución antes que la nueva agenda pueda ser implementada (previo al *itp*); la expresión (4), a aquéllas que se encontrarán en ejecución en dicho momento; y finalmente la expresión (5), identifica las tarea aún no ejecutadas. Cabe señalar que la identificación debe realizarse en el orden de presentación de las expresiones.

$$\forall\, pt_{b,u_f} | pt_{b,u_f}.start < itp \wedge pt_{b,u_f}.end \geq et \Rightarrow pt_{b,u_f}.status = cancelled \quad (1)$$

$$\forall\, pt_{b,u_f}, pt_{b,u} | pt_{b,u} \in pb_b \wedge pt_{b,u_f}.status = cancelled \Rightarrow pt_{b,u}.status = cancelled \quad (2)$$

$$\forall\, pt_{b,u}, pb_b | pt_{b,u} \in pb_b \wedge (pt_{b,u}.end < et) \vee (u \neq u_f \wedge et \leq pt_{b,u}.end < itp \wedge pt_{b,u}.status \neq cancelled) \Rightarrow pt_{b,u}.status = finished \quad (3)$$

$$\forall\, pt_{b,u} | u \neq u_f \wedge pt_{b,u}.start < itp \wedge pt_{b,u}.end \geq itp \wedge pt_{b,u}.status \neq cancelled \Rightarrow pt_{b,u}.status = inProcess \quad (4)$$

$$\forall\, pt_{b,u} | pt_{b,u}.start \geq itp \wedge pt_{b,u}.status \neq cancelled \Rightarrow pt_{b,\mathrm{u}}.status = toBeExecuted \quad (5)$$

Clasificación de los "batches" que componen la agenda de trabajo

A partir de la clasificación de las tareas programadas presentada, se puede realizar una clasificación de los distintos "batches" en aquéllos que se encuentran terminados (6), en procesamiento (7), a ser procesados (8) y cancelados (9).

$$\forall pt_{b,u}, pb_b | pt_{b,u} \in pb_b \wedge (\forall\, pt_{b,u}.status == finished) \Rightarrow pb_b.status = finished \quad (6)$$

$$\forall pt_{b,u}, pb_b | pt_{b,u} \in pb_b \wedge \exists\, pt_{b,u}.status == inProcess \wedge \nexists\, pt_{b,u}.status == cancelled \Rightarrow pb_b.status = inProcess \quad (7)$$

$$\forall pt_{b,u}, pb_b | pt_{b,u} \in pb_b \wedge (\forall\, pt_{b,u}.status == toBeExecuted) \Rightarrow pb_b.status = toBeExecuted \quad (8)$$

$$\forall pt_{b,u}, pb_b | pt_{b,u} \in pb_b \wedge (\forall\, pt_{b,u}.status == cancelled) \Rightarrow\ pb_b.status = cancelled \quad (9)$$

Clasificación de los "batches" según sus posibilidades de "rescheduling"

Esta clasificación está basada en la definición de los parámetros $fiep_s$ y $ciep_s$ dada por las expresiones (10) y (11). Los "batches" pueden clasificarse en aquéllos que son cancelados y, por ende, deberán reprocesarse (*tbr*, "to be reprocessed", expresión 12). También se identifican los "batches" que podrían llegar a ser reprocesados (*pr*, "possibly reprocessed", expresión 13). Estos son los que estando en ejecución poseen una tarea asignada al equipo que sale de funcionamiento durante el período de indisponibilidad del mismo, la cual tenía comienzo previsto luego del *itp*. Para estos "batches" podría darse el caso de que no haya ninguna posibilidad de reacomodar la tarea afectada de forma factible, si se respetan las secuencias de procesamientos prohibidas y/o las restricciones de espera correspondientes entre etapas. De igual manera se identifican los "batches" que no requieren ni podrán reiniciar su procesamiento, denominados *nrda*, "non-reprocessable directly affected" o *nr*, "non-reprocessable", (expresiones 14 y 15 respectivamente). La diferencia entre ellos radica en que a los primeros se les permite modificar su agenda de procesamiento, inclusive dentro del período de "freezing". En caso contrario, y dado que necesariamente se debe cambiar de unidad a la tarea afectada por una falla, podría no existir una agenda factible. Esta clasificación debe efectuarse en el orden en que se presentan las expresiones.

$$fiep_s = itp + apt_s \cdot fc, \forall\, s \in S \quad (10)$$

$$ciep_s = itp + apt_s \cdot fc + apt_s \cdot cc, \forall\, s \in S \quad (11)$$

$$\forall pb_b | pb_b.status == cancelled \Rightarrow pb_b.sched = tbr \quad (12)$$

$$\forall pt_{b,u}, pb_b | pt_{b,u} \in pb_b \wedge pt_{b,u_f}.status = toBeExecuted \wedge pt_{b,u_f}.start < urtp \wedge pb_b.status == inProcess \Rightarrow pb_b.sched = pr \quad (13)$$

$$\forall pt_{b,u}, pb_b | pt_{b,u} \in pb_b \wedge pt_{b,u_f}.start < urtp \wedge pb_b.status = = toBeExecuted \Rightarrow pb_b.sched = nrda \quad (14)$$

$$\forall pb_b | pb_b.sched \neq (tbr\ \vee pr \vee nrda) \Rightarrow pb_b.sched = nr \quad (15)$$

3.3 Modelo CP

El modelo CP que se presenta en esta sección fue implementado en el lenguaje OPL soportado por el paquete IBM ILOG CP Optimizer y el entorno de desarrollo IBM ILOG CPLEX Optimization Studio 12.4 [3]. Se ha optado por este ambiente por las siguientes facilidades para abordar problemas de "scheduling", especialmente del tipo reactivo: (i) inclusión de variables y constructores de alto nivel, (ii) rapidez para obtener buenas soluciones iniciales, (iii) capacidad para manejar eficientemente gran cantidad de variables, (iv) elevada flexibilidad en el modelado, generada a partir de la posibilidad de utilizar expresiones no lineales, entre otras.

Este modelo, que a diferencia del propuesto en [2] no surge de la definición de acciones de "rescheduling", toma como referencia los distintos intervalos descriptos en la sección 2 e ilustrados en la Figura 1. La restricción (16) establece que todas las tareas pertenecientes a los "batches" que fueron cancelados deberán comenzar su nuevo procesamiento una vez terminado el intervalo de "freezing" y que los que se iban a iniciar una vez concluido dicho período, no se adelanten. Similarmente, la restricción (17) establece la misma condición para los "batches" que fueron abortados.

Los "batches" cancelados son aquéllos que estaban siendo procesados en la unidad que falló, en el momento de la falla. Asimismo, si se trata de una política NIS/ZW, también se cancelarán aquéllos que debían comenzar su procesamiento en dicha unidad durante la ventana de "rescheduling". La diferencia de éstos con los "batches" abortados es que antes de ejecutar el modelo se sabe que deberán ser reprocesados. Por su parte, los "batches" abortados son aquéllos que el modelo decide interrumpir por no poder reacomodar las tareas que aún restan llevarse a cabo.

$$startOf\left(stTask_{b,s}\right) \geq fiep_S, \forall\, s \in S, \forall\, b \in B, \forall\, u \in U_s, \{pb_b.sched = tbr \vee (pb_b.sched = nr \vee pt_{b,u}.start \geq fiep_s)\} \quad (16)$$

$$ab_b == 1 \Rightarrow startOf\left(stTask_{b,s}\right) \geq fiep_S, \forall\, s \in S, \forall\, b \in B, pb_b.sched = pr \quad (17)$$

La restricción (18) establece que todos los "batches" que se asignen a la unidad que falló deberán procesarse luego de que la misma entre nuevamente en operación.

$$startOf\left(task_{b,u_f}\right) \geq urtp \cdot presenceOf\left(task_{b,u_f}\right), \forall\, b \in B \quad (18)$$

Las restricciones (19) y (20) establecen los tiempos de inicio y fin, así también como la unidad de procesamiento, de las tareas que se mantendrán sin cambios respecto de la agenda original. Por su parte, la restricción (21) fija el tiempo de inicio más temprano para las tareas clasificadas como *nrda*.

$$startOf\left(task_{b,u}\right) = pt_{b,u}.start, \forall\, u \in U, \forall\, b \in B, pb_b.sched = nr, pt_{b,u}.start \leq fiep_{s_u} \quad (19)$$

$$endOf\left(task_{b,u}\right) = pt_{b,u}.end, \forall\, u \in U, \forall\, b \in B, pb_b.sched = nr, pt_{b,u}.start \leq fiep_{s_u} \quad (20)$$

$$startOf\left(stTask_{b,s}\right) \geq itp, \forall\, s \in S, \forall\, b \in B, pb_b.sched = nrda \quad (21)$$

Las restricciones (22) y (23) corresponden a tareas que se identificaron como finalizadas o en progreso, pertenecientes a "batches" que podrían ser abortados. En caso de que no se aborte el "batch" en cuestión, las tareas asociadas mantienen su tiempo de inicio, de fin y la unidad de procesamiento.

$$ab_b == 0 \Rightarrow startOf(task_{b,u}) = pt_{b,u}.start, \\ \forall u \in U, \forall b \in B, pb_b.sched = pr, pt_{b,u}.status = (finished \vee inProcess) \quad (22)$$

$$ab_b == 0 \Rightarrow endOf(task_{b,u}) = pt_{b,u}.end, \forall u \in U, \forall b \in B, \\ pb_b.sched = pr, (pt_{b,u}.status = finished \vee pt_{b,u}.status = inProcess) \quad (23)$$

La restricción (24) establece qué "batches" cambian de unidad asignada. Por su parte, la restricción (25) permite cuantificar la anticipación o demora en el tiempo de inicio de procesamiento de un "batch" en una cierta etapa, en la nueva agenda, respecto a los tiempos previstos en el "schedule" original.

$$presenceOf(task_{b,u}) = 1 - cu_{b,u}, \\ \forall pt_{b,u}, \forall u, u' \in U, \forall b, b' \in B, u = u' \wedge b = b' \quad (24)$$

$$startOf(stTask_{b,s}) - pt_{b,u_s}.start = -as_{b,s} + ds_{b,s}, \\ \forall b \in B, \forall s \in S, \forall u \in U_s \quad (25)$$

Las restricciones (26) y (27) establecen el valor de las variables binarias asociadas a la anticipación o al atraso de las tareas, las que toman valor 1 si se produce un adelanto o una demora, respectivamente. La expresión (28) impide que la ejecución de un "batch" en una etapa determinada exhiba anticipación y atraso simultáneamente.

$$as_{b,s} \leq wMk \cdot asb_{b,s}, \forall b \in B, \forall s \in S \quad (26)$$

$$ds_{b,s} \leq wMk \cdot dsb_{b,s}, \forall b \in B, \forall s \in S \quad (27)$$

$$asb_{b,s} + dsb_{b,s} \leq 1, \forall b \in B, \forall s \in S \quad (28)$$

La restricción (29) establece la duración de las tareas de procesamiento cuando se utiliza una política de almacenamiento intermedio/operación UIS o NIS/ZW. En caso de emplearse una política NIS/UW o NIS/FW el operador = deberá ser reemplazado por ≥. Además, si la política es NIS/FW también se deberá incluir la expresión (30).

$$sizeOf(task_{b,u}) = bpt_{p,u} \cdot presenceOf(task_{b,u}), \forall u \in U, \forall p \in P, \forall b \in B_p \quad (29)$$

$$sizeOf(task_{b,u}) \leq (bpt_{p,u} + mwt_{p,u}) \cdot presenceOf(task_{b,u}), \\ \forall u \in U, \forall p \in P, \forall b \in B_p \quad (30)$$

Todo "batch" debe ser procesado en una y sólo una unidad por etapa (31)

$$alternative\ (stTask_{b,s}, all(u \in U_s)\ task_{b,u}), \forall s \in S, \forall b \in B \quad (31)$$

Para establecer la condición de precedencia entre las tareas que pertenecen a un mismo "batch" se plantea la restricción (32), la cual aplica cuando se considera una política de almacenamiento intermedio/operación NIS/ZW, NIS/FW o NIS/UW. En

caso de adoptarse una política UIS es necesario cambiar el constructor *endAtStart* por *endBeforeStart.*

$$endAtStart\left(stTask_{b,s}, stTask_{b,s'}\right), \forall\, b \in B, \forall\, s, s' \in S, s \neq Card(S), \quad Ord\,(s') = Ord\,(s) + 1 \tag{32}$$

La restricción (33) permite insertar los tiempos de "changeover" entre tareas sucesivas, al mismo tiempo que impide que dos tareas realizadas en un mismo equipo se ejecuten con algún solapamiento. Por su parte, la expresión (34) impide que ocurran secuencias de procesamiento prohibidas. Ambas emplean variables de secuencia, cuyos detalles se presentaron oportunamente en [4].

$$noOverlap(unitBatchSeq_u, cot_u), \forall\, u \in U \tag{33}$$

$$typeOfNext\left(unitBatchSeq_u, task_{b,u}\right) \neq p', \forall (p, p')\ \in Fp, \forall\, u \in U, \forall\, b \in B_p \tag{34}$$

Las expresiones (35) a (37) definen las penalidades que utiliza el modelo para realizar la reformulación de la agenda de trabajo. La expresión (37) requiere la definición del parámetro *abP*, dado por (38). Esta penalidad es muy elevada y con ella se pretende que un "batch" sea abortado sólo cuando no hay otra opción de "rescheduling" factible; *abP* en realidad sobreestima la penalidad de un escenario donde todas las tareas adelantan su inicio y cambian de unidad. $Ept_{b,u}$ es una función que toma valor igual a uno para aquellas tareas pertenecientes a la agenda original que satisfacen las condiciones de la sumatoria.

$$\begin{aligned} cuTP = & \sum_{\substack{\forall\, b\in B, \forall\, u\in U, s\neq S_{uf}, pb_b.sched=(pr\vee nrda) \\ itp\leq pt_{b,u}.start\leq fiep_{u_s}}} cu_{b,u}\cdot(1-ab_b)\cdot cuP1 + \\ & \sum_{\substack{\forall\, b\in B, \forall\, u\in U, pb_b.sched=nr \\ fiep_{u_s}<pt_{b,u}.start\leq ciep_{u_s}}} cu_{b,u}\cdot cuP2 + \\ & + \sum_{\substack{\forall\, b\in B, \forall\, u\in U, s\neq S_{uf}, pb_b.sched=(pr\vee nrda) \\ fiep_{u_s}<pt_{b,u}.start\leq ciep_{u_s}}} cu_{b,u}\cdot(1-ab_b)\cdot cuP3 \end{aligned} \tag{35}$$

$$\begin{aligned} tcTP = & \sum_{\substack{\forall\, b\in B, \forall\, s\in S, s\neq s_{u_f}, pb_b.sched=pr \\ itp\leq pt_{b,u}.start\leq fiep_s}} \left(\frac{as_{b,s}}{wMk}\cdot asP1 + \frac{ds_{b,s}}{wMk}\cdot dsP1\right)\cdot(1-ab_b) \\ & + \sum_{\substack{\forall\, b\in B, \forall\, s\in S, pb_b.sched=nr \\ fiep_{u_s}<pt_{b,u}.start\leq ciep_{u_s}}} \left(\frac{as_{b,s}}{wMk}\cdot asP2 + \frac{ds_{b,s}}{wMk}\cdot dsP2\right) \\ & + \sum_{\substack{\forall\, b\in B, \forall\, s\in S, s\neq s_{u_f}, pb_b.sched=pr \\ fiep_{u_s}<pt_{b,u}.start\leq ciep_{u_s}}} \left(\frac{as_{b,s}}{wMk}\cdot asP3 + \frac{ds_{b,s}}{wMk}\cdot dsP3\right)\cdot(1-ab_b) \end{aligned} \tag{36}$$

$$abTP = \sum_{b \in B} ab_b \cdot abP \tag{37}$$

$$abP = \sum_{\substack{\forall\, b \in B, \forall\, u \in U, pb_b.sched \neq cancelled, \\ pt_{b,u}.status = toBeExecuted}} Ept_{b,u} \cdot (cuP1 + asP1) \tag{38}$$

Las expresión (39) especifica el "Makespan" y la (40), la función multi-objetivo. Ésta utiliza el parámetro *wP*, cuya definición está dada por la expresión (41), que también apela a la función $Ept_{b,u}$. El parámetro *wP* contempla el peor escenario donde (i) todas las tareas cambian de unidad y se adelantan, (ii) todos los "batches" clasificados como *pr* son abortados y (iii) los clasificados como *nrda* se reprograman al final de la agenda.

$$Mk = \max\left(endOf\left(stTask_{b,s}\right)\right), \forall\, b \in B, \forall\, u \in U \tag{39}$$

$$FO = w \cdot \frac{cuTP + tcTP + abTP}{wP} + (1 - w) \cdot \frac{Mk - bMk}{wMk - bMk} \tag{40}$$

$$wP = abP + \sum_{\forall\, b \in B, pb_b.sched = pr} Ept_{b,u} \cdot abP \tag{41}$$

4 Resultados

El modelo propuesto se validó mediante la resolución de numerosos ejemplos, basados en un caso de estudio de "scheduling" predictivo abordado por otros autores (ver datos en el ejemplo 4 del trabajo [5]). Se considera un ambiente productivo conformado por 12 unidades de procesamiento, algunas de ellas operando en paralelo, pero no idénticas entre sí, distribuidas en 5 etapas productivas. El proceso opera bajo políticas de almacenamiento intermedio/espera de tipo NIS/ZW o NIS/FW. Se tienen en cuenta tiempos de "changeover" dependientes de la secuencia. La agenda original incluye 12 "batches" correspondientes a productos diferentes.

Para este caso de estudio se resolvieron diferentes ejemplos – que no se incluyen por falta de espacio – asociados a eventos de ruptura de equipos y arribo de nuevas órdenes, con diferentes combinaciones de políticas de operación de la planta. Los datos, resultados y los diagramas de Gantt se presentan en [6]. En esta sección sólo se describe un ejemplo donde falla la unidad 4 (etapa 2) y se supone que la planta opera con una política NIS/FW en la primera etapa, siendo NIS/ZW en el resto. Los parámetros empleados se muestran en la Tabla 1, mientras que la agenda original se presenta en la Fig. 2. En dicho "schedule" se ha marcado el instante en que ocurre la ruptura del equipo, así como el período durante el cual éste no estará disponible, afectando a tareas pertenecientes a cuatro "batches" provocando que uno sea cancelado y otro abortado. También se muestran los diferentes intervalos en los que se dividió el horizonte temporal.

Tabla 1. Parámetros y datos utilizados en el ejemplo

cuP1=10	*asP1*=7	*dsP1*=6	*abP*=765	*wMk*=1566	*et*=200	*cc*=3
cuP2=5	*asP2*=5	*dsP2*=5	*wP*=1530	*w*=0.7	*rtw*=20	*urtp*=580
cuP3=1	*asP3*=3	*dsP3*=2	*bMk*=1061	u_f=4	*fc*=1	

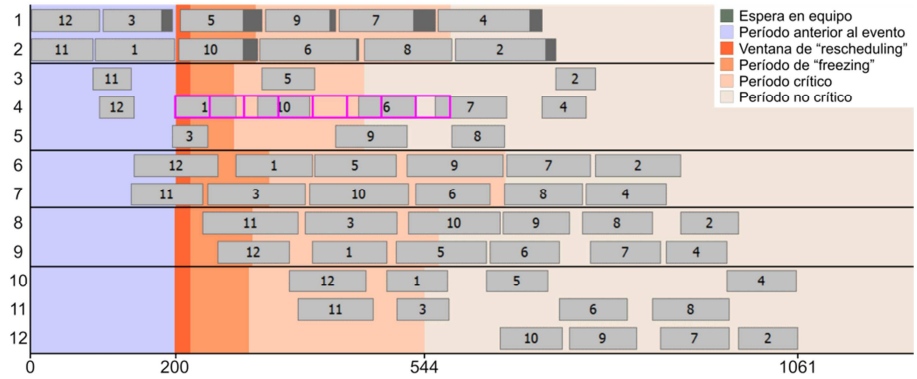

Fig. 2. Agenda de trabajo inicial

Como ya se mencionó, la penalidad relacionada a los "batches" abortados (*abP*) es muy grande con el objetivo de que los mismos se interrumpan definitivamente sólo cuando no hay otra opción factible. El resto de las penalidades presentadas en la Tabla 1 han sido seleccionadas de forma tal que para cada tarea, el cambio de equipo sea el tipo de modificación más disruptiva (penalidad más alta). Dentro de esta categoría de cambios, los menos deseados son los que se realizan en el intervalo de "freezing". Luego, en orden de importancia decreciente aparecen las modificaciones efectuadas en el "intervalo crítico" para "batches" indirectamente afectados por el evento. Finalmente, la menor penalidad se asigna a aquellos cambios de equipo realizados en el intervalo crítico para "batches" directamente afectados.

Cabe mencionar que se utilizaron criterios similares para seleccionar las penalidades asociadas a modificaciones en los tiempos de inicio. Dentro de esta gran categoría se fijó una penalización mayor para las tareas adelantadas respecto de las atrasadas, ya que se espera será más difícil responder a los cambios cuanto más pronto haya que realizarlos. La Tabla 2 resume la información relacionada con la solución encontrada, siendo P la suma de las penalidades totales *cuTP*, *abTP* y *tcTP*. Por su parte, la agenda de trabajo modificada se presenta en la Fig. 3.

Un análisis del "schedule" correspondiente a la solución obtenida muestra que en el intervalo de "freezing" no ocurre ningún cambio, en tanto en el intervalo crítico sólo se han agendado dos tareas pertenecientes al "batch" cancelado y hay una única tarea atrasada. Por el contrario, la mayor parte de las modificaciones, incluyendo las tareas del "batch" abortado, tienen lugar en el período no crítico. Asimismo, puede notarse que sólo dos tareas cambian la unidad asignada y un buen número de ellas se atrasa, pero lo hacen en el período no crítico. Este tipo de cambios, afortunadamente, no produce el denominado "nerviosismo" en el piso de planta.

Tabla 2. Parámetros de la solución hallada

FO: 0.43591	*Mk:* 1204	*P:* 767.0971	*cuTP:* 1	*abTP:* 765	*tcTP:* 1.0971

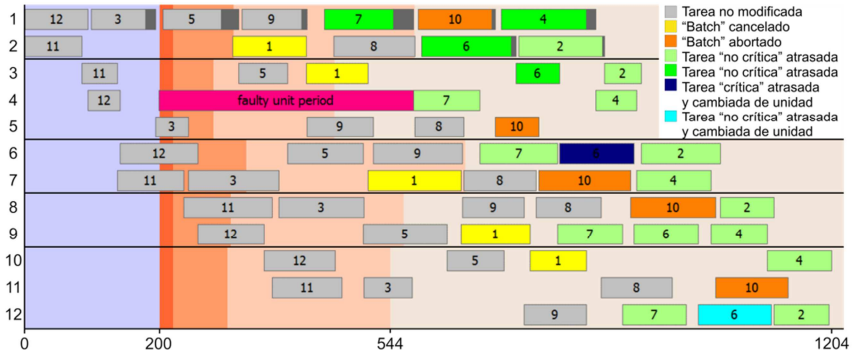

Fig. 3. Nueva agenda de trabajo

5 Conclusiones y comentarios finales

Se desarrolló un modelo CP para revisar agendas de trabajo de plantas "batch" multi-producto, multi-etapa afectadas por la salida de funcionamiento de una unidad o la llegada de una nueva orden. Dicho modelo no recurre al concepto de acción de "res-cheduling" utilizado en [2]. Asimismo, la propuesta emplea un novedoso enfoque multi-objetivo que trata de minimizar los cambios en el "schedule" inicial, a la vez que intenta mantener buenos valores del índice de desempeño originalmente emplea-do. El modelo ha sido probado en distintos escenarios. Como trabajo futuro se espera validarlo encarando ejemplos de mayor dimensionalidad (mayor número de órdenes a procesar y de equipos) y con más restricciones. Previo a ello se hará un análisis de sensibilidad de la formulación frente a variaciones en los parámetros elegidos

6 Referencias

1. Subramanian, K., Maravelias, C.T., Rawlings, J.B.: A state-space model for chemical pro-duction scheduling. Computers and Chemical Engineering 47, 97-110 (2012)
2. Novas, J.M., Henning, G.P.: Reactive scheduling framework based on domain knowledge and constraint programming. Computers and Chemical Engineering 34, 2129-2148 (2010)
3. IBM ILOG CPLEX Optimization Studio http://www-01.ibm.com/software/integration/ op-timization/cplex-optimization-studio/
4. Novara, F.M., Novas, J.M, Henning, P.G.: A comprehensive CP approach for the schedul-ing of resource-constrained multiproduct multistage batch plants. In: Computer-Aided Chemical Engineering 32, pp. 589 – 594. Elsevier (2013).
5. Marchetti, P.A., Cerdá, J.: A general resource-constrained scheduling framework for mul-tistage batch facilities with sequence-dependent changeovers. Computers and Chemical Engineering 33, 871-886 (2009)
6. Información de soporte, https://sites.google.com/site/NHreactivescheduling42JAIIO/