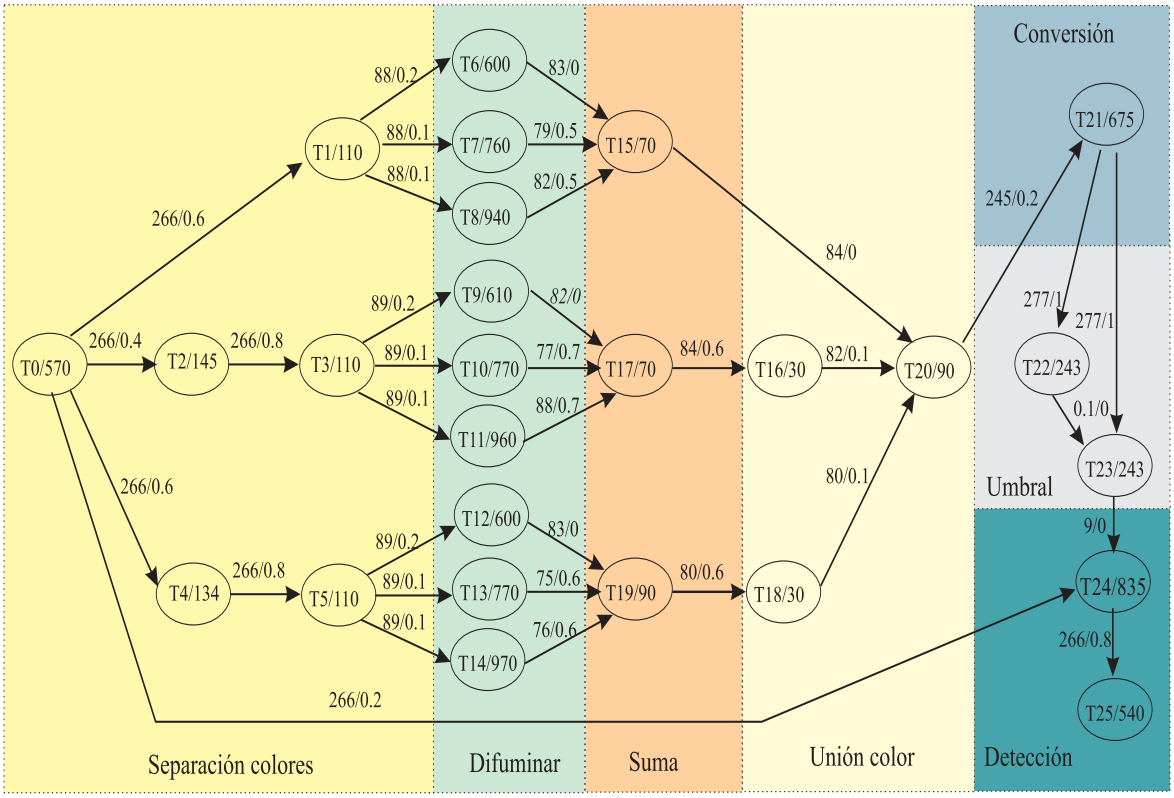

Figura 6.3: Grafo TTIG de la aplicación BASIZ. Tiempo de cómputo en milisegundos y volumen de comunicación en Kbytes.

2. *TIG mapping.* En el caso de utilizar el grafo TIG como modelo de la aplicación, se ha usado la heurística CREMA [SRCL98], que proporciona buenos resultados en comparación con otros algoritmos de mapping de la literatura basados en este mismo modelo [SRCE97].

   Como todas las políticas de mapping basadas en el modelo TIG, el principal objetivo de CREMA es balancear el cómputo y minimizar las comunicaciones. Para ello la estrategia se desarrolla en dos etapas donde primero se agrupan las tareas en clusters y luego se asignan los clusters a procesadores bajo el criterio de la función de coste *minimax*. El objetivo de la función *minimax* consiste en minimizar el máximo *PWL (Processor Work Load)* de la asignación final, donde el *PWL* de cada procesador se define como el coste total de cómputo más comunicación de todas las tareas asignadas a él.

3. *TPG mapping.* La asignación se basa en el modelo TPG de la aplicación. En particular se ha integrado la heurística ETF (Earliest Task First) [HCAL89].

   El mecanismo básico con el que se desarrolla el ETF es el siguiente. A cada iteración del algoritmo se computa el mínimo tiempo de inicio de cada tarea preparada en cada procesador. Se consideran tareas preparadas aquellas que todas sus predecesoras han sido ya ejecutadas. La tarea con mínimo tiempo de inicio es la que se asigna. Dentro de los algoritmos de scheduling de TPGs, éste proporciona unos resultados eficientes a costa también de una complejidad relativamente alta [KA99].

6.3 Resultados experimentales

Para la ejecución de las aplicaciones en el cluster se ha utilizado la herramienta AMEEDA (Automatic Mapping for Efficient Execution of Distributed Applications) desarrollada en nuestro grupo [Yan01] [YRR+02]. AMEEDA es una herramienta de mapping de propósito general que proporciona un entorno amigable para ejecutar de forma automática las aplicaciones de paso de mensajes aplicando una política de mapping concreta. En la Figura 6.4 se muestra el

diagrama de bloques de AMEEDA. La función de cada uno de los módulos se resume a continuación.

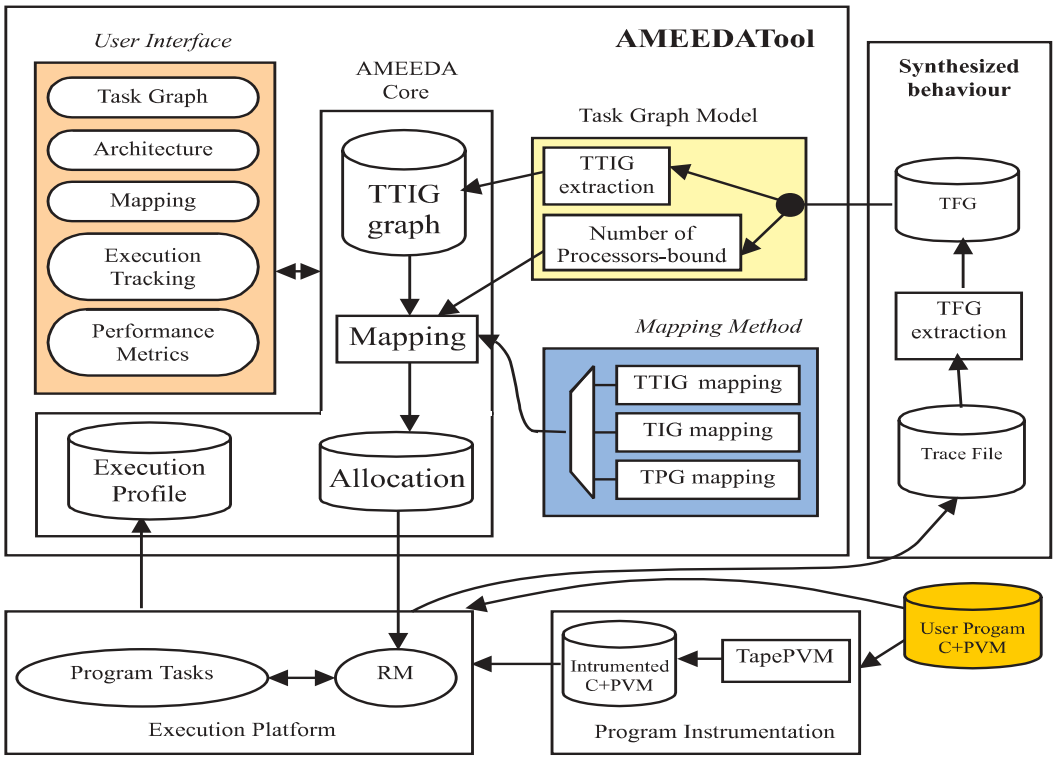

Figura 6.4: Diagrama de bloques de la herramienta AMEEDA.

- *Modelo de grafo de tareas.* A partir del grafo TFG obtenido de la aplicación con alguno de los métodos descritos en el Apartado 2.2, se deduce el modelo TTIG.

- *Método de mapping.* En AMEEDA están integradas las políticas de mapping basadas en los modelos TTIG, TIG y TPG descritas en el apartado anterior, para que el usuario pueda escoger la que considere más propicia en función del comportamiento de la aplicación. El mapping físico de tareas a procesadores, y su posterior ejecución se realiza de manera automática, basándose en la política de mapping seleccionada.

- *Interfase de usuario.* Este módulo proporciona un conjunto de opciones a través de ventanas que facilitan el uso de la herramienta.

En el Apéndice C se completa la descripción de las facilidades que proporciona AMEEDA y el entorno de ventanas que posee.

Mediante el uso de AMEEDA se ha ejecutado cada una de las aplicaciones descritas en al Apartado 6.1, en el cluster de PC's, variando el número de procesadores de 1 a 8.

Para cada aplicación y cada número de procesadores se han utilizado distintos algoritmos de mapping para comparar la efectividad de las asignaciones basadas en el modelo TTIG con respecto a las asignaciones basadas en los modelos clásicos TIG y TPG.

La bondad de las políticas basadas en el modelo TTIG se evalúa según el porcentaje de ganancia, %G, obtenido en el tiempo de ejecución respecto a una política de los modelos clásicos, y se calcula con la siguiente expresión:

$$\%G = \frac{T_{CLA} - T_{TTIG}}{T_{CLA}} \times 100 \tag{6.1}$$

donde:

T_{CLA} corresponde al tiempo de ejecución obtenido al usar la política de mapping que corresponde a cada uno de los modelos clásicos.

T_{TTIG} corresponde al tiempo de ejecución obtenido al ejecutar la aplicación con una de las políticas basadas en el modelo TTIG.

En el Apéndice D se muestran los valores de tiempo de ejecución obtenidos al ejecutar cada aplicación con cada política concreta. A continuación se realiza el análisis de los porcentajes de ganancia para las aplicaciones con comportamiento arbitrario y las aplicaciones con comportamiento homogéneo respectivamente.

6.3.1 Aplicaciones con comportamiento arbitrario

En la Figura 6.5 se muestra en sendos diagramas de barras el porcentaje de ganancia obtenido para las políticas TASC y MATE con respecto a CREMA, para las aplicaciones con comportamiento arbitrario y topología irregular AP1,..,AP5.

Observando los resultados de las figuras se puede ver que en general las dos

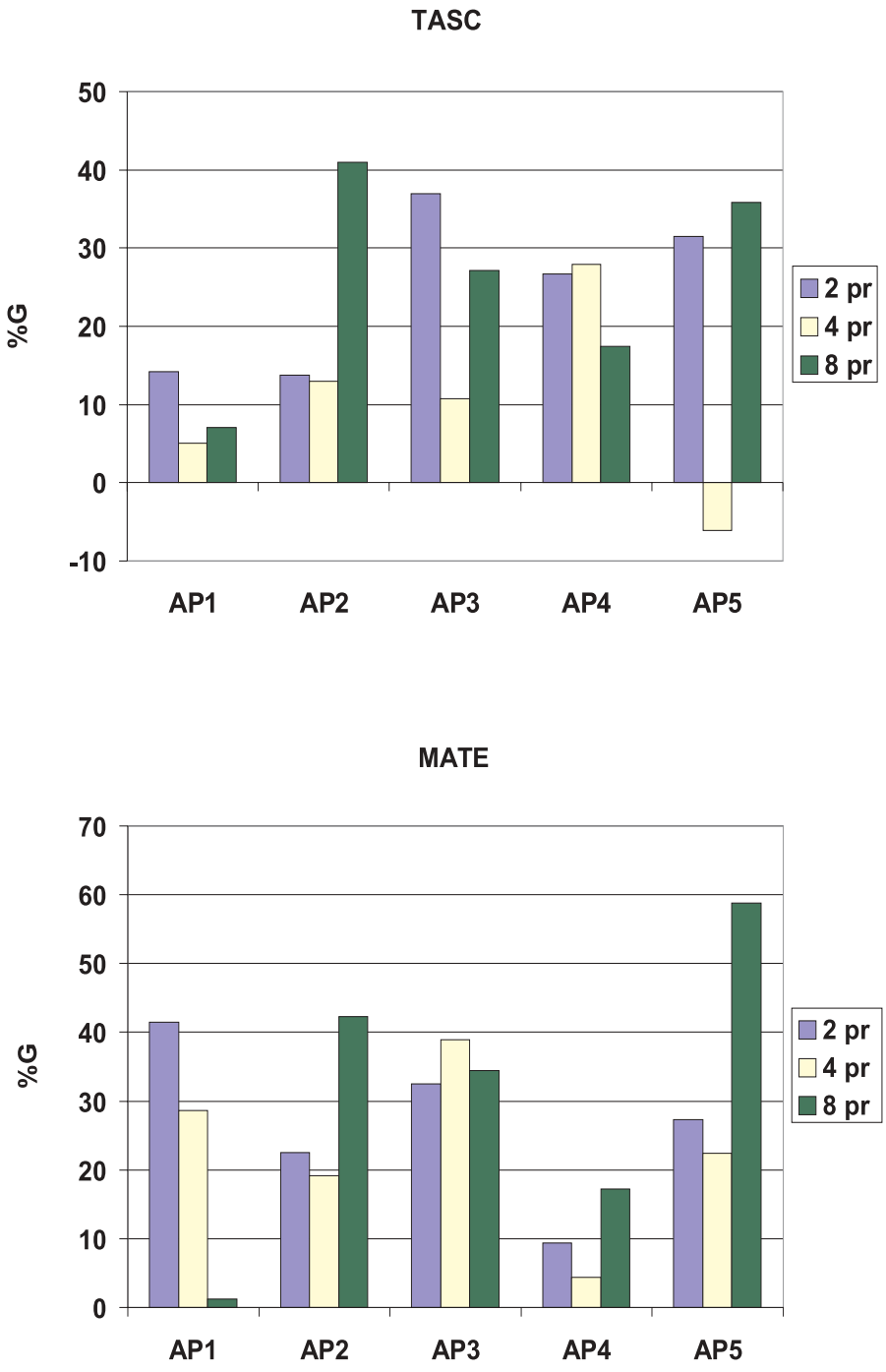

Figura 6.5: Ganancia de TASC y MATE respecto CREMA para las aplicaciones con comportamiento arbitrario y topología irregular.

políticas TASC y MATE proporcionan unos valores de ganancia importantes respecto a CREMA (cabe exceptuar únicamente el caso de AP5 con cuatro procesadores en la política TASC), lo que continua corroborando un buen comportamiento de las estrategias de mapping basadas en el modelo TTIG.

Si se comparan los resultados de los dos diagramas de barras entre sí, vemos que exceptuando la aplicación AP4, en todos los casos la estrategia MATE proporciona mayor valor de ganancia que la política TASC. Este hecho continua siendo consistente con el análisis realizado para ambas políticas en el Capítulo 5 al comparar con el óptimo.

Para las aplicaciones con comportamiento arbitrario y estructura regular se han evaluado únicamente los valores de ganancia de la política MATE respecto CREMA, puesto que se vio en el Capítulo 5 que ambas estrategias, TASC y MATE, tienen un comportamiento prácticamente igual en este tipo de topologías. En la Figura 6.6 se muestra el porcentaje de ganancia obtenido al variar el número de procesadores utilizados en la asignación.

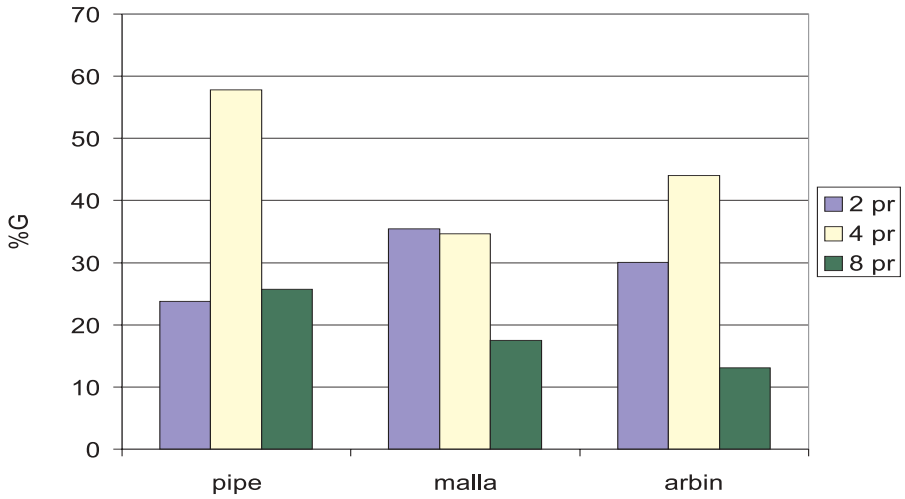

Figura 6.6: Ganancia de MATE respecto CREMA para las aplicaciones con comportamiento arbitrario y topología regular.

A la vista de los resultados se continua poniendo de manifiesto en este caso la efectividad de usar una política de mapping basada en un modelo capaz de capturar las diferencias en el comportamiento temporal de las tareas. Vemos

que en todas las topologías se dan unos valores de ganancia importantes que llegan a ser del 58% para la topología pipe con cuatro procesadores.

6.3.2 Aplicaciones con comportamiento homogéneo

Para las aplicaciones con comportamiento homogéneo, puesto que están basadas en topologías regulares, se ha medido únicamente la ganancia de la estrategia MATE con respecto a CREMA para los distinto grados de paralelismo.

En el caso de las aplicaciones con paralelismo bajo, puesto que su comportamiento es cercano al de un TPG, se ha evaluado también la ganancia obtenida respecto a la estrategia ETF. En la Figura 6.7 se muestran los resultados de ganancia obtenidos al comparar con ambas políticas.

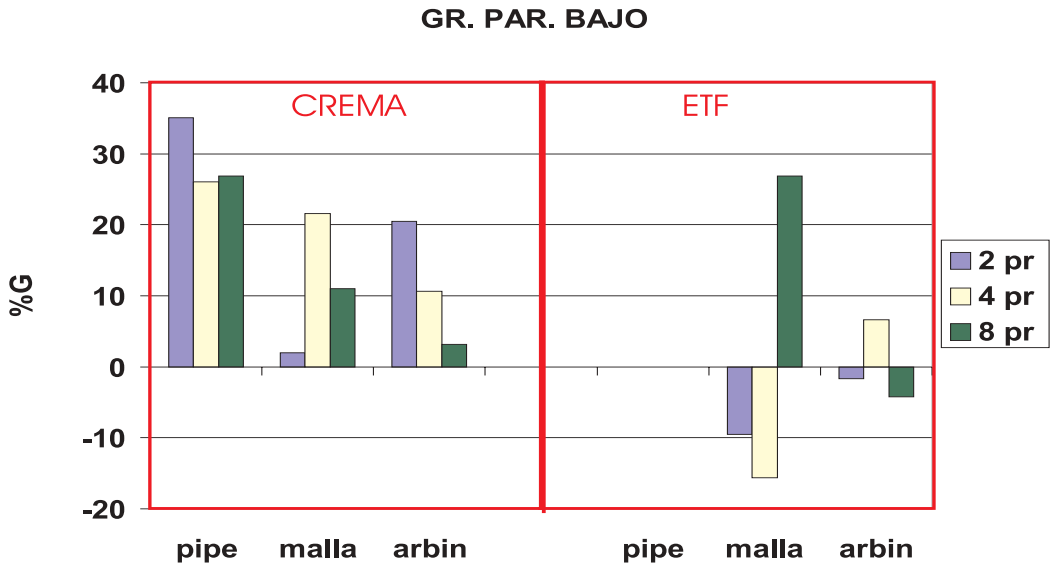

Figura 6.7: Ganancia de MATE respecto CREMA y ETF para las aplicaciones con comportamiento homogéneo y grado de paralelismo bajo.

En el caso de CREMA vemos que se obtienen valores de ganancia positivos en todos los casos. No obstante, respecto la estrategia ETF, para la topología pipe se obtiene una ganancia igual a 0, puesto que tanto MATE como ETF coinciden en la asignación, en cambio para las topologías malla y árbol binomial se obtienen tanto ganancias positivas como negativas. Teniendo en cuenta

que la estrategia ETF está perfectamente probada y testeada para los grafos TPG [KA99], esto no hace más que darnos una indicación de la bondad de la estrategia MATE, puesto que es capaz de generar resultados comparables a ETF cuando se trata de aplicaciones con un comportamiento muy cercano al del modelo TPG.

Para las aplicaciones con paralelismo medio y alto, la Figura 6.8 muestra los resultados de ganancia de MATE respecto CREMA.

Cuando se trata de grado de paralelismo medio, MATE siempre da valores de ganancia positivos respecto CREMA, que en algunos casos puntuales son muy elevados (véase topología arbin). En cambio, cuando se trata de aplicaciones con paralelismo alto, se obtienen tanto ganancias positivas como negativas. Esto es debido a que dichas aplicaciones tienen el mismo comportamiento que se asume en el modelo TIG (todas las tareas simultáneas), y por lo tanto la estrategia CREMA constituye de por sí una buena política de asignación [SRCE97]. Vemos también en este caso que la política basada en el modelo TTIG es capaz de generar resultados comparables a una estrategia específica del modelo, cuando el comportamiento de la aplicación es el que se asume en el modelo TIG.

6.3.3 Aplicación BASIZ

La aplicación BASIZ se ha ejecutado para una secuencia de 20 imágenes de entrada. Se ha probado experimentalmente que este es el número de imágenes que proporciona un mejor *throughput* en la estructura pipeline.

En este caso se ha realizado la asignación con las políticas MATE, CREMA y ETF, basadas en los modelo TTIG, TIG y TPG respectivamente. Además, se ha ejecutado la aplicación para la asignación *round-robin* que es la que implementa por defecto PVM. Respecto a las políticas de mapping utilizadas, es preciso remarcar que el algoritmo ETF se ha aplicado para estudiar los resultados que se obtienen en el tiempo de ejecución si la aplicación se modelase como un grafo TPG (es decir, asumiendo que todos los grados de paralelismo son 0), aunque evidentemente esta no es la situación real de comportamiento de la aplicación.

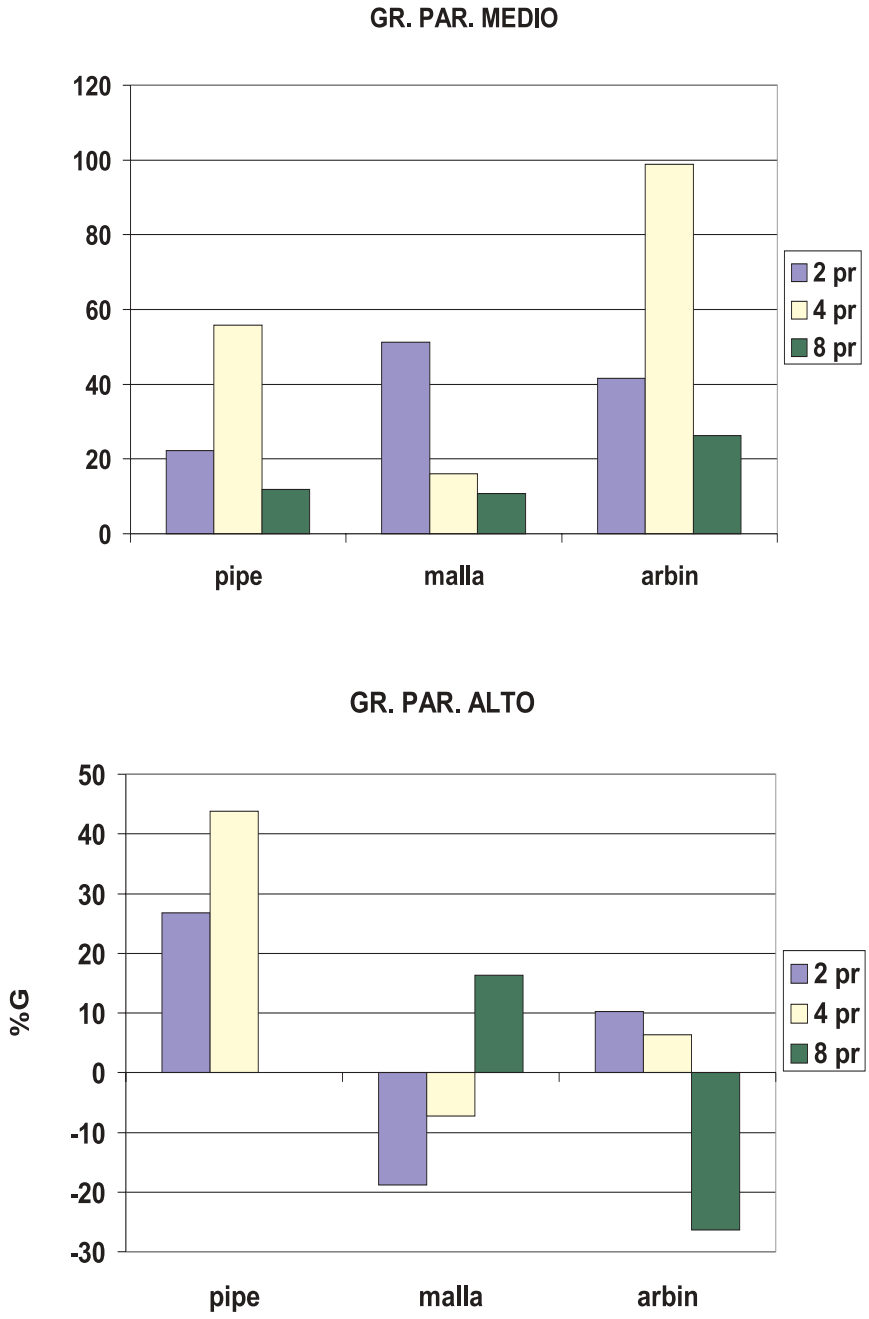

Figura 6.8: Ganancia de MATE respecto CREMA para las aplicaciones con comportamiento homogéneo y grado de paralelismo medio y alto.