Tarea 4

Características Principales CFS:

1. **Equidad**: CFS asigna proporciones de tiempo de CPU a cada tarea en función de su "nice value", que varía de -20 (alta prioridad) a +19 (baja prioridad). Esto permite una distribución justa del tiempo de CPU.
2. **Uso de `vruntime`**: En lugar de asignar prioridades fijas, CFS utiliza un valor de tiempo de ejecución virtual (`vruntime`) para determinar cuál tarea debe ejecutarse a continuación. Tareas con menor `vruntime` tienen prioridad.
3. **No slices de tiempo discretos**: CFS no usa cortes de tiempo fijos. En cambio, se basa en una latencia objetivo que define el tiempo que cada tarea debe ejecutarse, ajustándose dinámicamente al número de tareas activas.
4. **Clases de programación**: Linux utiliza diferentes clases de programación (CFS para tareas normales y clases de tiempo real) para adaptar el algoritmo a las necesidades del sistema.
5. **Balanceo de carga**: CFS implementa un balanceo de carga que toma en cuenta la utilización de la CPU y las prioridades de las tareas, minimizando la migración de hilos entre núcleos.
6. **Consciencia de NUMA**: En sistemas NUMA, CFS organiza los núcleos en dominios de programación para optimizar el acceso a la memoria, evitando migraciones que podrían aumentar la latencia.

Funcionamiento:

1. **Selección de tareas**: CFS elige la tarea con el menor `vruntime` para ejecutarse. Si una tarea de mayor prioridad se vuelve elegible, puede interrumpir a una tarea de menor prioridad.
2. **Ejecución de tareas I/O y CPU**: Las tareas que son I/O-bound tienden a tener un `vruntime` menor, lo que les permite recibir más tiempo de CPU que las tareas CPU-bound.
3. **Políticas de tiempo real**: CFS también admite tareas de tiempo real mediante políticas SCHED_FIFO y SCHED_RR, que tienen prioridad sobre las tareas normales.
4. **Balanceo dentro de dominios**: El balanceo de carga se realiza inicialmente dentro de un mismo dominio de programación, evitando movimientos entre núcleos de diferentes dominios para minimizar la latencia de acceso a memoria.

En resumen, CFS busca proporcionar un uso equitativo y eficiente de los recursos de CPU en sistemas Linux, adaptándose a diferentes tipos de carga de trabajo y condiciones del sistema.