

## Práctica 5 HPC (GitHub) Vladimir Estal Daries

Esta práctica me ha parecido interesante y completa. En general todo bastante didáctico y factible, aunque he tenido unos problema que expongo:

En el script tuve un problema con los comentarios, ya que me salían duplicados. El problema de los comentarios duplicados se debe a que se están ejecutando múltiples instancias del script en paralelo debido a la configuración `#SBATCH --ntasks-per-node=4`. Esto significa que cada instancia del script produce su propia salida de comentarios. Para que los comentarios solo aparezcan una vez en la salida, hice uso de la variable de entorno `SLURM_PROCID` proporcionada por SLURM. Esta variable contiene el ID del proceso actual dentro de la tarea en ejecución.

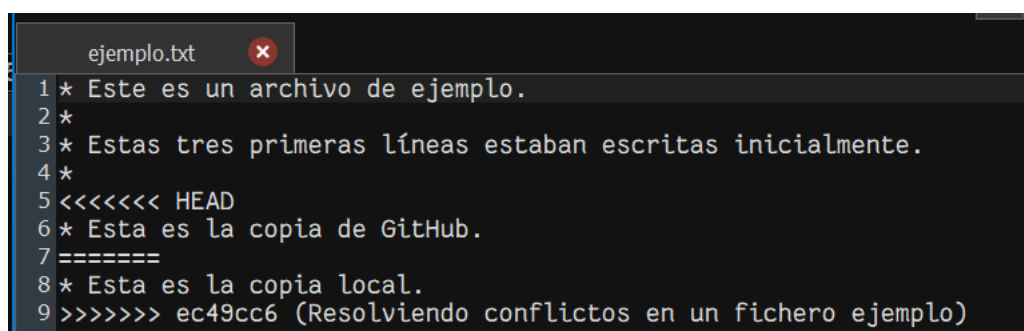
```
if [ "$SLURM_PROCID" == "0" ]; then
```

```
fi
```

El bloque de código dentro del `if [ "$SLURM_PROCID" == "0" ]` se ejecutará solo en la primera instancia del script, mientras que las instancias posteriores no ejecutarán ese bloque de código. Esto garantiza que los comentarios solo se impriman una vez en la salida, independientemente de la cantidad de instancias que se estén ejecutando en paralelo.

Algo que he aprendido, es que si copias un directorio que es un repositorio de GitHub a otro directorio en tu sistema local, conservarás todo el historial de control de versiones y podrás trabajar con él con normalidad utilizando Git y GitHub. Cuando copias el directorio, estás copiando todos los archivos y subdirectorios que forman parte del repositorio, incluyendo el directorio oculto `.git`, que es donde se almacena toda la información relacionada con el control de versiones. Esto significa que todos los commits, ramas, etiquetas y configuraciones asociadas con el repositorio también se copiarán. Una vez que hayas copiado el repositorio a otro directorio, podrás usar los comandos de Git normalmente dentro de ese directorio para hacer cambios, crear commits, sincronizar con GitHub, entre otros.

El mayor problema lo tuve justo en el último paso, cuando había que realizar el merge. Cuando modifiqué el archivo en GitHub me equivoqué nombrando el commit. Como debían de tener el mismo nombre de commit ambos archivos, intenté borrar el commit directamente en la web GitHub, pero esa opción no está disponible. Por lo que me informé, debían de coincidir el historial del repositorio local y el remoto para poder modificar los commit desde la terminal, así que tras fusionarlos intenté utilizar `git reset --hard <commit-hash>` mediante la terminal utilizando `git log` para identificar el hash pertinente. Después usé `git push --force origin main` y continué con el ejercicio, pero cuando intenté llevar a cabo el merge (ver imagen abajo) me daba error que me impedía actualizar los cambios en el repositorio remoto y ya no supe solucionarlo sin volver a iniciar todo de nuevo.



```
ejemplo.txt x
1 * Este es un archivo de ejemplo.
2 *
3 * Estas tres primeras líneas estaban escritas inicialmente.
4 *
5 <<<<<< HEAD
6 * Esta es la copia de GitHub.
7 =====
8 * Esta es la copia local.
9 >>>>>> ec49cc6 (Resolviendo conflictos en un fichero ejemplo)
10
```

Por esto considero que se debería de profundizar más en este aspecto de GitHub, ya que estoy seguro de que puede ser una importante fuente de problemas especialmente para usuarios inexpertos.