TP2: Empaquetador de Tornillos

Santiago Alvarez Juliá, *Padrón Nro. 99522*
75.42 Taller de Programación
Facultad de Ingeniería, Universidad de Buenos Aires

Abril 2018

Índice general

0.1. Introducción

En las siguientes secciones abordaré algunos temas claves para la resolucion del programa Empaquetador de Tornillos. Respecto a la base de programacion de ejercicios anteriores, este difiere en 2 puntos claves, centrales en la programación:

- POO (Programación Orientada a Objetos)
- Concurrencia

0.2. Temas Claves

0.2.1. POO

Excepto la funcion main, todo el resto del programa fue diseñado en base a objetos que se relacionan entre si. A continuación haré una breve descripcion de cada objeto de la aplicación.

- Empaquetador : su objetivo es inicializar el programa en si. Abre todos los archivos que llegan como parametros a la funcion main y los amacena en un TDA para su posterior uso. Tambien se encarga de realizar el informe final de remanentes y de inicializar el objeto Packages que encapsula al TDA vector de ScrewPackage (ambos objetos serán descriptos individualmente posteriormente). Pero su principal funcion es la de lanzar 1 hilo (es decir, crear otro objeto) por cada archivo de clasificacion .bin.
- Clasificador : es el objeto creado por Empaquetador que hereda de la clase Thread (que es simplemente una clase que encapsula std::thread) que sobreescribe el metodo run. Clasificador se encarga de leer todos los archivos de clasificación (procesar las tuplas de 4 bytes) y con la informacion procesada de los tornillos que deben agregarse a sus respectivos paquetes, llama al metodo addScrews de Packages.
- Packages : el objeto Packages encapsula 2 TDA map : el primero almacena ScrewPackages con la clave el id de tornillo y el segundo almacena mutexes (propios de concurrencia) con la clave id de tornillo. Su funcion es agregar los tornillos a los paquetes y se encarga de la concurrencia del programa a traves de los mutex.

- ScrewPackage: se encarga de encapsular algunas propiedades claves de lo que seria un paquete de tornillos: el id de los tornillos que almacena, su capacidad, la cantidad de tornillos dentro y un vector ordenado de ints que representan el ancho de cada tornillo almacenado. Sus metodos principales son addScrews y getMediana (se refiere a la media del ancho de tornillos). Para el calculo de la mediana es necesario almacenar los ints en orden, por eso elijo el TDA vector que es simple y es ordenado facilmente con std::sort.

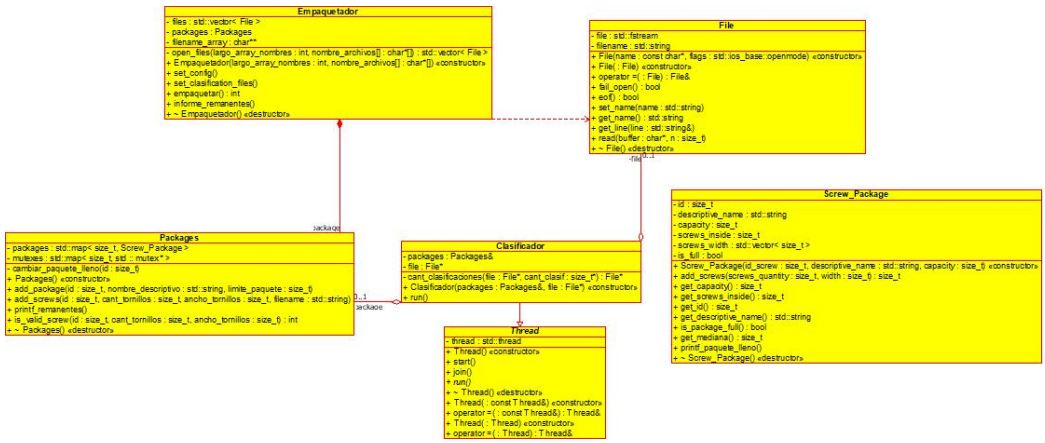

Figura 1: Diagrama de Clases de Empaquetador de Tornillos

0.2.2. Concurrencia

Una manera simple de definir a la concurrencia es la separacion de tareas en distintos hilos. Para eso (Packages hereda de threads(cambiar)).

Dentro del metodo Empaquetar de la clase Empaquetador se lanzan los hilos, 1 por cada archivo de clasificación ya que la información que almacenan puede ser procesadada independientemente. El mayor problema en la concurrencia de este programa sucede cuando se quiere modificar o pedir informacion de los paquetes y cuando se quiere imprimir por pantalla (tambien conocidos como Race Conditions).

- Paquetes : cada paquete tiene su propio mutex ya que agregar tornillos en distintos paquetes son funciones independientes entre si. Dichos mutex son lockeados cuando se agregan tornillos para evitar problemas de concurrencia. Dentro de AddScrews se verifica si se lleno el paquete y en el caso de que asi sea, se cambio por un nuevo objeto Paquete

vacio y se imprime por pantalla que se lleno el Paquete anterior, por lo tanto se evitan problemas como agregar tornillos en un paquete que fue previamente llenado en otro hilo.

- Print por pantalla : antes de lanzar los hilos se abren los archivos de clasificacion y se imprime por pantalla si fue correcta su apertura. Luego de "joinear"los hilos se imprime el informe de remanentes.

  Mientras se ejecutan los hilos pueden imprimirse por pantalla 2 cosas distintas: un error en la informacion del tornillo y cuando se llena un paquete. Para imprimir errores utilizo una clase que se llama Fprintf-Protected que tiene su propio mutex como atributo, que es al principio de cada uno de sus metodos. Cuando se llena un paquete, se imprime por pantalla que paquete se lleno y cual es la mediana del ancho de tornillos. Como se llenan paquetes al agregar tornillos (AddScrews), se lockea el mutex propio del paquete antes de imprimir por pantalla.