Mejoras

Este documento es una revisión de las mejoras propuestas en la primera entrega y cómo estas han cambiado para la segunda entrega. También se incluyen las mejoras por hacer en la tercera entrega.

# Mejoras propuestas en la primera entrega

De las mejoras que se propusieron en la primera entrega se ha realizado principalmente la mejora del algoritmo de creación. A pesar de las distintas maneras en las que pensábamos que podíamos corregir el algoritmo de creación, a la hora de implementarlas, muchas han presentado problemas que no hemos sabido resolver.

Como ya se ha comentado en la primera entrega, el algoritmo de creación provisional que se entregó fue un parche de última hora y gran parte del tiempo de esta segunda entrega lo hemos tenido que balancear entre el diseño de capas y la finalización de lo que iba a ser nuestro algoritmo de creación. ¿Que cantidad de recursos (tiempo y personas) podíamos permitirnos invertir? Como se puede entender fácilmente, no es una decisión fácil, decidimos crear una deadline artificial que diera tiempo a un equipo de nuestros integrantes para intentar tener una versión mínimamente funcional del nuevo algoritmo de creación. Nos tenía a todos bastante entusiasmados, porque refinaba mucho más el proceso de creación.

Por desgracia esta deadline artificial llego y los resultados seguian teniendo carencias. El proceso de debugar se hacía muy costoso, y a pesar de que sabíamos donde fallaba el código, no conseguimos establecer con seguridad cómo se podían obtener resultados dada una segunda deadline artificial, y hubo una moción unánime por parte del grupo de no dar un voto de confianza al nuevo algoritmo. Fue un momento difícil porque suponía que mucho trabajo y mucho código no “vería la luz del día”, pero esperamos que las lecciones aprendidas transpiren a continuación.

Que aprendimos del algoritmo de creación:

* Las secuencias pueden servir para detectar que casillas blancas no pueden tener candidatos.
* El algoritmo de resolución puede retornar los posibles candidatos reales de cada casilla blanca. Información que podría usarse para elegir otras sumas totales para los padres.
* El algoritmo de creación ha de entrelazar la generación con el testeo, sino, en este tipo de problemas se pueden alargar demasiado.

# Mejoras propuestas para la tercera entrega

Como ya hemos mencionado, la mayor parte del tiempo lo estamos dedicando al correcto funcionamiento del proyecto, no obstante, no nos conformamos con el algoritmo de creación que tenemos actualmente, es por eso que para la tercera entrega, si bien no vamos a lograr tener ese algoritmo tan óptimo que planteamos al inicio del proyecto, tenemos toda una serie de mejoras concretas sobre el trabajo ya hecho. Si bien estas mejoras no suponen ningún tipo de cambio fundamental en la estructura de los algoritmos, creemos que esto es una muy buena señal, porque confirma que mediante la revisión hemos encontrado mejoras particulares que enriquecen los algoritmos, pero los fundamentos los consideramos sólidos. (importante a esta altura del proyecto)

Las mejoras para cada algoritmo son parcialmente las ya comentadas en la primera entrega, sobre todo respecto al algoritmo de resolución. Su correcto funcionamiento nos da mucha garantía de su calidad y eficiencia, y mejorarlo es de una prioridad muy baja. Pero las mejoras siguen estando por hacer. Principalmente revisar ciertas condiciones necesarias pero no suficientes en el proceso de backtracking. Estos serían una serie de comprobaciones polinomiales que esperamos reduzcan el coste medio de considerar una permutación como inválida, ya que es en el caso que sean inválidas que normalmente experimentamos el coste peor. Como es aparente estas mejoras son útiles pero no modifican el caso mejor o peor, siendo mejoras inteligentes pero no cruciales.

En el caso del algoritmo creación, debido a que finalmente usaremos el algoritmo provisional, hay muchos más frentes que mejorar. En primer lugar, debido a su carácter provisional, no fue sometido a escrutinio suficiente y a posteriori hemos detectado errores graves en la función principal. A grandes rasgos se trata de un bucle que en el caso de que el tablero (compuesto por casillas negras y blancas sin números ni sumas asignados) esté mal formado o no tenga ninguna combinación que sea solución, simplemente no termina. Este error es muy fácil de corregir y es una mejora de alta prioridad para la tercera entrega. Otros errores menores ya han sido corregidos para la segunda entrega, pero se espera una tercera iteración de debugging en grupo, que es cuando el equipo al completo revisa el algoritmo.

En segunda instancia, el algoritmo provisional tiene 2 secciones importantes, una en que se crea la distribución del tablero y una segunda en que se encuentran valores de las sumas para cada tira que son correctas. La segunda parte simplemente tiene que ser revisada pero tiene un comportamiento esperado y no esperamos encontrar grandes mejoras a implementar. Pero es la primera sección la que tiene que recibir todo un set de mejoras importantes. En su estado actual es capaz de “dibujar” 2 plantillas de kakuros, como fue explicado en la primera entrega, pero la mejora que debemos implementar es tener un algoritmo capaz de crear plantillas o distribuciones con más complejidad. Esto supone un gran avance respecto a nuestro primer intento de hacer el algoritmo de creación y nos permite aplicar muchos conceptos que ya teníamos para esta implementación, pero ejecutados en un entorno mucho más modular y compartimentalizado, que está ayudando mucho a que las distintas partes del código sean más estancas e independientes, y entonces, más fáciles de trabajar.