

A ordenar, cada cosa en su lugar: una experiencia de migración a la virtualidad de contenidos sobre algoritmos de ordenamiento en la universidad

Darío Ocles¹, Pablo Turjanski^{1,2}[0000-0002-6009-9650], and Matías Lopez-Rosenfeld^{1,2}[0000-0003-4254-1481]

¹ Universidad de Buenos Aires. Facultad de Ciencias Exactas y Naturales. Departamento de Computación. Buenos Aires, Argentina.
docles,pturjanski,mlopez@dc.uba.ar

² CONICET-Universidad de Buenos Aires. Instituto de Investigación en Ciencias de la Computación (ICC). Buenos Aires, Argentina.

Resumen Con el advenimiento de la pandemia de COVID-19, los contenidos de la materia Algoritmos y Estructura de Datos I de la FCEyN-UBA -que habían sido pensados previamente para ser dictados de manera presencial-, fueron adaptados para ser dictados bajo una modalidad virtual. En este trabajo se presentan algunos de los cambios realizados en la clase de “Algoritmos de ordenamiento sobre secuencias”. En particular, se detalla la experiencia de incorporar un videojuego y preguntas asociadas a estos, para que los estudiantes experimenten y reflexiones sobre la temática durante distintos momentos de la clase. Los algoritmos que se presentan en la clase son *selection sort*, *insertion sort* y *counting sort*. Tanto las partidas como las respuestas a las preguntas del videojuego fueron registradas y analizadas. Como resultado preliminar se pudo observar que la estrategia más utilizada por los estudiantes para ordenar fue la misma que el algoritmo *selection sort*. Esto se observó incluso antes de la presentación del algoritmo. En cuanto a las preguntas realizadas, se detectaron términos en las respuestas que indican cierto grado de incorporación de los temas y apropiación de la terminología. Para finalizar, y en base a la experiencia realizada, concluimos que ciertos aspectos que acercó la virtualidad deberán ser tenidos en cuenta al momento de retornar a la presencialidad.

Keywords: Algoritmos · Ordenamiento · Virtualidad · Didáctica.

1. Introducción

Inicios del año 2020. Como todos los años, una nueva edición de la materia Algoritmos y Estructuras de Datos I (AED1) en la Facultad de Ciencias Exactas y Naturales (FCEyN) de la Universidad de Buenos Aires (UBA) está a punto de comenzar. Sin embargo, un evento inesperado obliga a cancelar el comienzo

de su dictado: *pandemia*. Se hace imposible el dictado de la materia en su modalidad presencial. La única alternativa es reestructurar la materia para realizar su dictado en una modalidad virtual. Este cambio no sólo significa modificar su contenido sino también cambios en el plantel docente (docentes que abandonan la materia por falta de una buena conexión a internet en sus domicilios o que se toman licencia, docentes que se incorporan a la materia debido a que otras materias no pueden ser dictadas a distancia, etc.), en la infraestructura (aula físicas por salas de videoconferencias), etc. Todas estas modificaciones se dieron en un lapso de tiempo muy breve y condicionaron el dictado, pero a pesar de ello se pudo dar la materia en una modalidad virtual incorporando ciertas herramientas que, en su modalidad presencial, no hubiesen sido ni siquiera pensadas. En particular en este trabajo, haremos mención a la experiencia de transformar a la virtualidad uno de los temas de la materia: “Algoritmos de ordenamiento sobre secuencias”.

Esta transformación va en línea con la búsqueda permanente, del equipo docente de la materia, de involucrar a los estudiantes en una experiencia de aprendizaje activa, metodología que ha mostrado interesantes resultados en experiencias previas [10,11,13]. En este caso fue materializado, principalmente, a través de un videojuego en el que es posible explorar el problema de ordenar. Opinamos que es una de las mejores formas de incorporar los conceptos detrás de un tema, haciendo, participando y siendo parte de él [2].

En lo que sigue, el trabajo se organiza de la siguiente manera. En la sección 2 se detalla la materia, y en particular la temática que compartimos en este manuscrito; en la sección 3 se detalla el videojuego y particularidades de la implementación en versión virtual; en la sección 4 se muestran los resultados obtenidos; finalmente en la sección 5 se realiza una discusión acerca de los resultados obtenidos, debatiendo sobre las potencialidades y nuevas dimensiones posibles en los contextos virtuales.

2. La materia y la temática

AED1 es el primer contacto con una materia puramente de computación que tienen los alumnos de la carrera de la Licenciatura en Ciencias de la Computación de la FCEyN, UBA. Los temas introducidos van desde la especificación de problemas hasta alcanzar su resolución a través de la algoritmia, haciendo foco principalmente en definir sin ambigüedades cuál es el problema a resolver (el *qué*) para finalmente abordar la manera de resolverlo (el *cómo*). Esto se ve plasmado tanto en la bibliografía sugerida por el equipo docente (ver Tabla 1) como en el orden en que se dictan los temas en el curso. Se podría discutir si para una materia introductoria el enfoque es el correcto o no, pero esto excede el objetivo de este manuscrito.

Las clases de AED1 (tanto en la modalidad presencial como en la modalidad a distancia) se encuentra divididas en dos categorías i) clases teóricas y ii) clases prácticas. En las primeras se introducen los temas desde un abordaje teórico,

Cuadro 1: Bibliografía utilizada en la materia AED1, FCEyN, UBA.

Título	Autores	Referencia
The science of programming	David Gries	[7]
Reasoned Programming	Krysia Broda y colegas	[1]
Introduction to algorithms	Thomas Cormen y colegas	[4]

mientras que en las segundas se realizan resoluciones de ejercicios prácticos tanto de manera individual como grupal (taller).

En el caso de las clases teóricas modalidad virtual, son presentadas una vez por semana abordando un tema nuevo cada vez. Para ello, se les indica a los estudiantes que vean de manera asincrónica una serie de videos cortos (de entre 10 y 15 minutos cada uno), intercalados por preguntas acerca del video observado. La finalidad del cuestionario es que los estudiantes puedan reafirmar la comprensión de los contenidos vistos. Tanto la visualización de los videos como las preguntas se encuentran implementadas en el campus, que utiliza la plataforma Moodle [5], de la FCEyN, UBA. Las clases teóricas se desarrollan en el siguiente orden:

- Clase 1: Especificación de problemas
- Clase 2: Secuencias
- Clase 3: Corrección y Teorema del Invariante
- Clase 4: Precondición más débil
- Clase 5: Precondición más débil de ciclos
- Clase 6: Buenas prácticas de programación
- Clase 7: Testeo Estructural
- Clase 8: Algoritmos de búsqueda sobre secuencias
- Clase 9: **Algoritmos de ordenamiento sobre secuencias**
- Clase 10: Algoritmos sobre secuencias ya ordenadas
- Clase 11: Algoritmos de búsqueda de un patrón en un texto
- Clase 12: Verificación automática de programas

2.1. Algoritmos de ordenamiento sobre secuencias

En el presente trabajo nos centraremos en el tema “Algoritmos de ordenamiento sobre secuencias”, dictado en el contexto de la teórica en la Clase 9. Previo a la pandemia, este tema se trataba en una clase teórica expositiva, en un aula *común*, dentro de la cual se realizaban simulaciones y seguimientos en el pizarrón en donde se los invitaba a los estudiantes a participar proponiendo métodos, enunciando estrategias y discutiendo acerca de sus ventajas y limitaciones. Si bien solían ser clases participativas, como suele suceder en un curso de entre 60 y 80 estudiantes, sólo una proporción muy pequeña de los estudiantes era la que participaba activamente (algunos no participaban por timidez, otros porque sus ideas eran similares a las expuestas por otros, etc.).

En la modalidad virtual decidimos cambiar la estrategia de enseñanza de esta clase teórica. El principal cambio que incorporamos fue el de *proponer a cada*

uno de los estudiantes enfrentarse y explorar por sus propios medios el problema y dar una propuesta de solución a través de un videojuego.

La implementación de la clase y sus actividades las llevamos a cabo a través del campus utilizando un módulo *lesson* provisto por Moodle, dedicado en su totalidad a la clase *Algoritmos de ordenamiento sobre secuencias*. El módulo *lesson* permite incorporar una secuencia de contenidos y preguntas intercaladas. La secuencia diseñada se divide en seis grandes secciones: comienza con 1) un trabajo previo a la clase; luego 2) se presenta una experiencia participativa de ordenar una secuencia de elementos desordenados; a continuación se introducen los algoritmos de ordenamiento 3) *selection sort*; 4) *insertion sort*; y 5) *counting sort*; para finalmente realizar 6) un cierre de clase donde se pone en juego lo aprendido. En la Tabla 2 se describe con mayor detalle la secuencia en cada una de estas secciones.

Las experiencias participativas a las que los estudiantes son invitados son distintas variaciones del problema de ordenar cartas de manera creciente utilizando un videojuego especialmente diseñado para tal fin, cuya única operación para resolverlo es intercambiar las cartas de 2 posiciones (no hay lugares extras, ni se pueden tomar de más de 2 cartas a la vez).

En la primera etapa, previo a comenzar con el contenido del tema, invitamos a cada estudiante a que ordene una secuencia desordenada de cartas (ver Fig. 1). El objetivo detrás de esta actividad es doble: i) hacer que el estudiante se familiarice con el problema de ordenar una secuencia de elementos desordenados previo a comenzar la clase (que identifique diferentes aspectos sobre el problema: su dificultad, casos especiales, posibles soluciones, etc.), y ii) registrar el método de ordenamiento de secuencias desordenadas que utiliza cada estudiante previo a ver los temas. En una segunda etapa, comenzamos con el dictado de la clase, introduciendo los algoritmos de ordenamiento *selection sort*, *insertion sort* y *counting sort*. Luego de la presentación de los 2 primeros les solicitamos a cada estudiante que ordene una secuencia de cartas utilizando el algoritmo recientemente aprendido en una adaptación del videojuego inicial que restringe los movimientos para que sólo se pueda avanzar con el ordenamiento si se aplica el algoritmo propuesto. No contamos con un videojuego para *counting sort* ya que no aplica en su lógica la operación de intercambio de posiciones para ordenar. En la etapa final, le pedimos al estudiante a repetir la experiencia del inicio de la clase. El objetivo que hay detrás de esta última actividad es darle la oportunidad de una última exploración para afianzar los temas visto, a la vez que capturamos este comportamiento para indagar si el estudiante se apropia de alguno de los métodos de ordenamiento introducidos durante la clase y, en caso de hacerlo, si mejora la eficiencia (por ejemplo, si disminuye la cantidad de movimientos que necesitó para ordenar la secuencia).

3. Materiales y Métodos

Los resultados del presente trabajo se basan en los datos obtenidos de los juegos y las preguntas relacionadas al videojuego detallados en la Tabla 2. Es-

Cuadro 2: Secuencia de actividades correspondiente al módulo *lesson* de la clase teórica 09: *Algoritmos de ordenamiento sobre secuencias*, AED1, FCEyN, UBA.

Sección	Descripción
1. Inicio de la clase	<ul style="list-style-type: none"> i) Video Bienvenida y presentación del tema ii) Videojuego ordenar una secuencia de cartas inicialmente desordenada iii) Preguntas relacionadas al videojuego
2. Introducción al ordenamiento	<ul style="list-style-type: none"> i) Video de introducción al problema de ordenar ii) Preguntas sobre el video previo
3. Algoritmo <i>selection sort</i>	<ul style="list-style-type: none"> i) Video introduciendo el método <i>selection sort</i> ii) Preguntas sobre el video previo iii) Videojuego ordenar una secuencia de cartas desordenada utilizando <i>selection sort</i> iv) Preguntas relacionadas al videojuego v) Serie de videos y preguntas acerca de la implementación, corrección y complejidad del algoritmo <i>selection sort</i>
4. Algoritmo <i>insertion sort</i>	<ul style="list-style-type: none"> i) Video introduciendo el método <i>insertion sort</i> ii) Preguntas sobre el video previo iii) Videojuego ordenar una secuencia de cartas desordenada utilizando <i>insertion sort</i> iv) Preguntas relacionadas al videojuego v) Serie de videos y preguntas acerca de la implementación, corrección y complejidad del algoritmo <i>insertion sort</i>
5. Algoritmo <i>counting sort</i>	<ul style="list-style-type: none"> i) Videos y preguntas acerca de la implementación, corrección y complejidad del algoritmo <i>counting sort</i>
6. Cierre de la clase	<ul style="list-style-type: none"> i) Video de cierre repasando los temas abordados durante la clase y preguntas para reafirmar lo visto ii) Videojuego mismo problema que en el inicio de la clase: ordenar una secuencia de cartas inicialmente desordenada iii) Preguntas relacionadas al videojuego iv) Bibliografía utilizada y sugerida para ampliar lo visto

6 D. Ocles et al.



Figura 1: Videojuego de ordenamiento de cartas utilizado en clase. El estudiante debe ordenar las cartas en orden creciente, de izquierda a derecha, utilizando como único movimiento posible el intercambio (*swap*) entre dos de ellas. Se puede observar la notación de arreglos sobre las cartas en la parte superior. El objetivo de esto es que el estudiante identifique la posición de cada carta, y afiance el sistema posicional al momento de pensar en el algoritmo de ordenamiento.

pecíficamente, se registró la totalidad de los movimientos realizados en el **videojuego** por los estudiantes que cursaron en ambos cuatrimestres de 2020 la clase “Algoritmos de ordenamiento sobre secuencias” (ver puntos 1.ii, 3.iii, 4.iii y 6.ii de la Tabla 2). Las respuestas a las preguntas posteriores (ver puntos 1.iii, 3.iv, 4.iv y 6.iii de la Tabla 2) fueron registradas en el campus y posteriormente extraídas a una planilla de cálculo de manera manual. En la Tabla 3 se puede observar un detalle de las preguntas realizadas.

3.1. Videojuego

El videojuego fue implementado en Javascript [3] utilizando el framework CreateJS [8]. Los datos correspondientes a los movimientos fueron almacenados en una base de datos MongoDB [9] y accedidos utilizando el framework Flask [16] sobre Python3 [17]. Para asociar los movimientos realizados en el videojuego con las respuestas a las preguntas posteriores del módulo *lesson*, se solicitó a cada estudiante que en el módulo *lesson* ingrese un identificador producido por el videojuego cada vez iniciado un nuevo juego. Algunas partidas han sido descartadas por errores en la transcripción de dicho identificador por parte de los alumnos.

3.2. Selección de la secuencia inicial a ordenar

Como secuencia inicial del videojuego siempre se presentó $I=[10, 8, 2, 6, 5, 4]$. La razón para presentar siempre la misma secuencia fue poder analizar

Cuadro 3: Preguntas realizadas luego del cada videojuego de la clase teórica 09 *Algoritmos de ordenamiento sobre secuencias*, AED1, campus FCEyN, UBA.

Sección	Preguntas
1. Previo al comienzo de clase	¿Siguieron una estrategia en particular? ¿Cuál? ¿Piensan que existe más de una manera de ordenar la secuencia? ¿Cuáles?
3. Algoritmo <i>selection sort</i>	¿Cuán parecidos son los movimientos que se hacen en este algoritmo a los que hicieron en el primer ordenamiento?
4. Algoritmo <i>insertion sort</i>	¿Cuán parecidos son los movimientos que se hacen en este algoritmo a los que hicieron en el primer ordenamiento?
6. Posterior a la clase	La relación de la estrategia que usaste recién con la que usaste en el primer desafío, ¿son iguales o distintas? Si cambiaste algo, ¿por qué lo cambiaste? ¿qué fue lo que te hizo cambiar? Si no cambiaste la estrategia, ¿hay algo de la clase que te sirva para reforzar la decisión que tomarste al ordenar?

el desempeño de los estudiantes bajo una misma condición inicial. Para evitar que los juegos sean visualmente similares, se alternó el palo de las cartas (**corazones, diamantes, tréboles y picas**) entre las distintas instancias de juego, sin embargo se restringió a que en una misma instancia todas las cartas sean del mismo palo.

El orden inicial de las cartas (I) no fue seleccionado al azar, sino que se eligió una que tuviera una carta en su posición final (el 6 en nuestro caso) y las demás por acomodar. De esta manera se presentó una *configuración* (posible orden de las cartas) que puede ser ordenada en un mínimo de 4 intercambios (recordemos que la operación que permitimos es dadas 2 posiciones, intercambiar las cartas). En la Fig. 2 se pueden observar, representadas como rectángulos, la totalidad de las posibles configuraciones por las que puede pasar un estudiante para ir desde la configuración inicial hasta la final utilizando exactamente 4 intercambios de cartas. En la figura se observan 42 configuraciones posibles. Con flechas se representan sus movimientos posibles. En particular, hay 125 secuencias de intercambios (camino) que llevan desde la configuración inicial propuesta a la final en 4 movimientos. En particular, sólo una de éstas representa la secuencia de intercambios del algoritmos *selection sort* (denotada en la figura con flechas verdes).

En cada instancia donde jugaron se les permitió a los estudiantes poder volver a jugar nuevamente. En caso de hacerlo, se les genera una secuencia inicial nueva al azar.

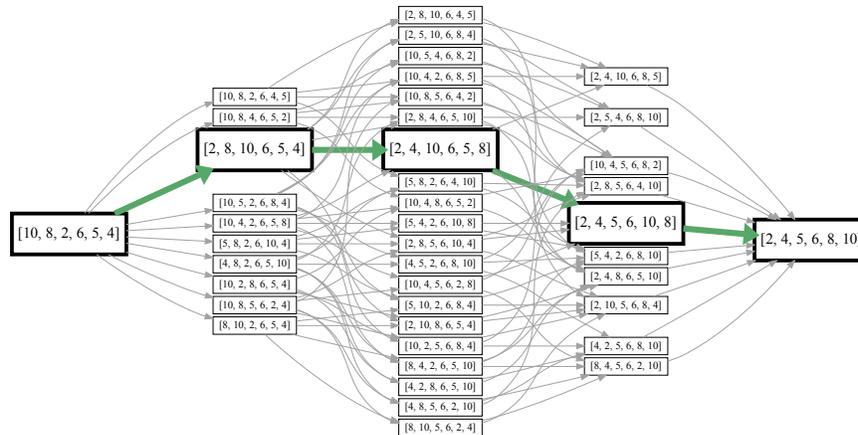


Figura 2: Caminos posibles para ordenar las cartas, realizando la cantidad de movimientos mínimos (4 intercambios de cartas) desde la configuración inicial hasta la final. En verde se observa el camino realizado por el algoritmo *selection sort*.

3.3. Análisis de los datos y representación gráfica

Para el análisis de los datos del presente trabajo conservamos por cada estudiante su primer intento finalizado cada vez que jugó cada modo del videojuego. Por finalizado nos referimos a aquel juego en que alcanza la configuración final (las 6 cartas ordenadas). Jugadas posteriores no fueron contempladas. Se excluyeron partidas con registros inconsistentes que impedían su reconstrucción (menos del 1% de los registros).

Para las representaciones gráficas y el análisis de los datos se utilizó Python3 [17] con las bibliotecas Seaborn [18] para los gráficos de barras y los gráficos de relaciones; GraphViz [6] para la exploración de trayectorias seguidas por los estudiantes al momento de ordenar la secuencia de cartas; WordCloud [14] para las nubes de palabras; y para la exclusión de palabras vacías (*stopwords* en inglés), utilizamos las *stopwords* del paquete NLTK [12] en español. En WordCloud se incluyeron solamente las 200 palabras más utilizadas. Ante la posibilidad de que se vea disminuida la frecuencia de ciertos términos, se eliminaron las tildes de las palabras (ej. é por e) y se reemplazaron letras especiales del alfabeto (ej. ñ por n)

4. Resultados

Entre el primer y segundo cuatrimestre de 2020 un total de 182 estudiantes participaron, al menos una vez, de los 4 modos del videojuego. En la Fig. 3 se

puede observar cómo se distribuyen en función de la versión del videojuego jugado. A medida que avanza la clase se observa una disminución en la cantidad de estudiantes que juegan el videojuego en sus diferentes modos. Esto es esperable ya que algunos estudiantes (pocos) comienzan la clase, pero no la finalizan.

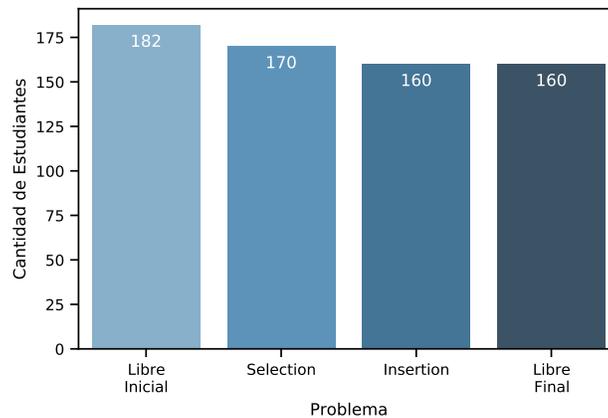


Figura 3: Cantidad de estudiantes que finalizaron los modos *libre inicial*, *selection sort*, *insertion sort* y *libre final* del videojuego.

Como primer etapa del análisis, comparamos la cantidad de movimientos que realizó cada estudiante entre el primer modo de videojuego y el último. Los resultados se pueden observar en la Fig. 4. Notar que esta figura sólo tiene en cuenta a aquellos estudiantes que han finalizado ambos juegos. El 66% de los estudiantes no alteran la cantidad de jugadas. En su gran mayoría suelen ser los estudiantes que realizan la cantidad óptima de movimientos (4 movimientos). El 19% de los estudiantes tiene una peor performance (aumentan la cantidad de movidas), y un 15% mejora su performance. La comparación con otros modos del videojuego no se ha realizado ya que no son las mismas condiciones del juego (ej. *selection sort* e *insertion sort* advierten al estudiante en caso de un movimiento incorrecto).

En una segunda etapa analizamos los movimientos realizados por los estudiantes previa y posteriormente a tomar la clase (modo libre inicial y libre final, respectivamente). En la Fig. 5 se puede observar un esquema similar al utilizado en la Fig. 2. En este caso las configuraciones (recuadros) mostradas corresponden a las diferentes *configuraciones* utilizadas por los estudiantes para llegar desde el ordenamiento inicial $I=[10,8, 2, 6, 5, 4]$ (rojo) hasta el final $F=[2, 4, 5, 6, 8, 10]$ (verde). La columna en la que se encuentra ubicada cada configuración, coincide con la cantidad mínima de intercambios entre pares de cartas que habría que hacer desde dicha configuración para llegar a F. En el caso de I, se encuentra ubicada a 4 columnas de la configuración F. El grosor de las flechas

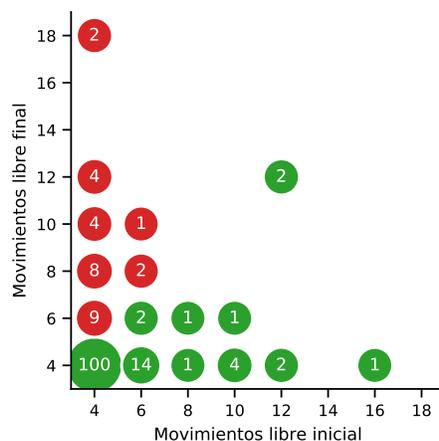
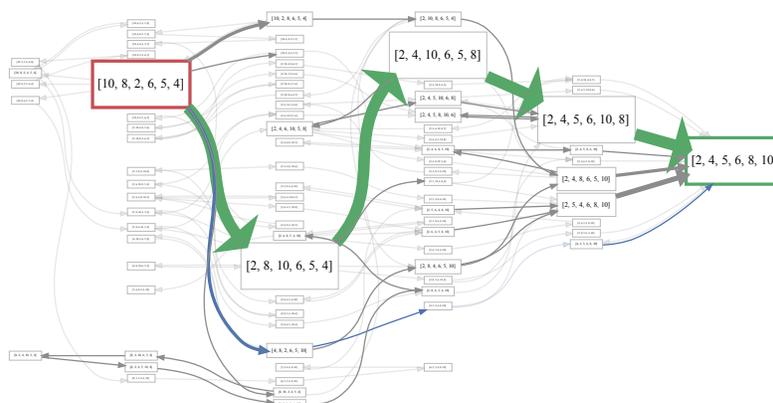


Figura 4: Relación entre los movimientos realizados por los estudiantes entre los modos videojuego *libre inicial* y *libre final*. En rojo los estudiantes que aumentaron la cantidad de movimientos posteriormente a haber realizado la clase. En verde aquellos que mantuvieron la cantidad o la disminuyeron.

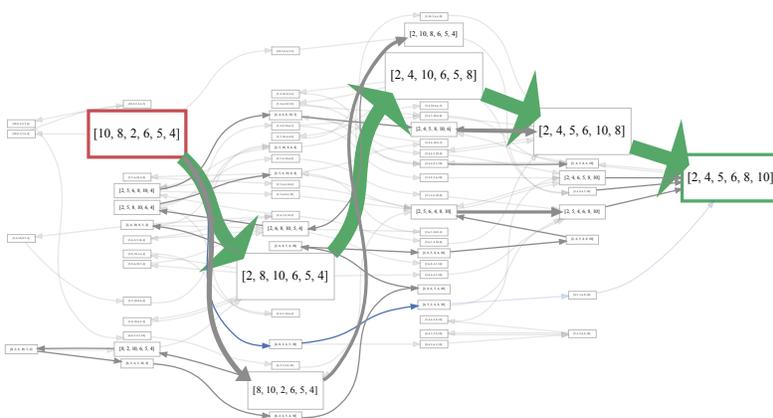
es proporcional con la cantidad de estudiantes que realizaron ese pasaje entre configuraciones. De color verde se encuentran resaltadas las flechas correspondientes a los pasos que realiza el algoritmo *selection sort*. En azul, se colorearon las flechas correspondientes a aplicar una variante del algoritmo *selection sort*, reubicando siempre el elemento máximo en vez del mínimo. Se puede observar que en el modo libre inicial los estudiantes exploran cerca de un 20% más de configuraciones que en el libre final (82 en libre inicial vs. 69 en libre final, ver. Fig. 5). La mayoría de los estudiantes, en ambos casos, elige el camino seguido por el algoritmo *selection sort*. Esta tendencia aumenta en el modo libre final. Por último, también se puede observar que muy pocos estudiantes se alejan de la solución final F (ver configuraciones ubicadas en la columna a la izquierda de la configuración I).

Por último, realizamos un análisis de las respuestas a las preguntas realizadas luego de jugar al modo libre inicial y final. Para ello generamos nubes de palabras con las respuestas (ver Figura 6). En el caso de libre inicial, la pregunta que realizamos fue relacionada a si siguieron una estrategia en particular, y en tal caso, cuál siguieron (ver Sección 1 de la Tabla 3). En la nube de palabras aparecen términos como *carta*, *mas*, *menor*, *chico*, *busque*, *puse*, *numero*, *izquierda*, *principio*. Estos términos parecerían referirse a la estrategia seguida por los estudiantes. En el caso de libre final, la pregunta estaba relacionada a si la estrategia seguida en esta ocasión era similar o no a la utilizada en el modo de videojuego libre inicial (ver Sección 6 en la Tabla 3). En esta oportunidad aparecen los términos *algoritmo*, *estrategia*, *selection*, *sort* y *cambie*. Una de las limitaciones de esta técnica es el caso de la palabra *cambie*, que en las respuestas

A ordenar, cada cosa en su lugar... 11



(a) Libre inicial



(b) Libre final

Figura 5: Movimientos realizados por los estudiantes en los modos del videojuego libre inicial y libre final. Cada rectángulo representa una configuración de las cartas. La inicial está marcada en rojo, la final en verde. Los ejes representan la transición entre dos configuraciones (el intercambio de 2 cartas). El tamaño indica la cantidad de estudiantes que pasaron por ese camino. Las flechas verdes indican el camino que sigue el algoritmo *selection sort*, mientras que las azules indican el camino de un *selection sort* que toma el mayor en lugar del menor.

frecuentemente se encontraba asociada a la negación *no*, sin embargo la palabra *no* no aparece en la nube ya que fue eliminada explícitamente por pertenecer al conjunto de *stopwords*. De sus 34 usos del término *cambie*, 13 indicaron un cambio, mientras que 21 indicaban lo contrario.

es natural y armonioso con la lectura occidental [15], sino que además uno de los trayectos más frecuentemente recorridos coincide con los pasos que sigue el algoritmo *selection sort* (ver Fig. 5a, flechas verdes). Surge de este punto un interrogante importante para futuras investigaciones, ¿es efectivamente el algoritmo *selection sort* una forma natural que tenemos las personas adultas para ordenar? ¿es cultural? ¿o es acaso producto de las limitaciones impuestas por el ambiente donde se realiza el ordenamiento?

Otra pregunta que nos hicimos fue si los estudiantes modificaban la manera de ordenar las cartas luego de tomada la clase. Para ello comparamos la cantidad de movimientos realizados en el modo de videojuego libre inicial contra el del libre final. Nuestra hipótesis es que si el cambio es muy grande (muchos estudiantes aumentan o disminuyen la cantidad) estamos en presencia de un indicador que habla de un cambio en la manera de ordenar. En la Fig. 4 se puede observar que la gran mayoría no modifica la cantidad de movimientos (104 de 160 estudiantes), con lo cual no hay evidencia de que cambien de estrategia. En la misma figura se indican en verde cuántos mantuvieron (o disminuyeron) la cantidad de movimientos, mientras que en rojo se pueden ver cuántos aumentaron la cantidad al realizar el segundo intento. Si bien la proporción de los que reducen o mantienen no es insignificante, es importante mencionar que no era la consigna optimizar la cantidad de pasos.

Al analizar los caminos recorridos en el modo del videojuego libre inicial (ver Fig. 5a) observamos que la cantidad de configuraciones exploradas es mayor que en el libre final. Esto nos podría estar indicando una mayor tendencia a explorar posibilidades por parte de los estudiantes, lo que se relaciona con una mayor experiencia vivida en la resolución del problema. Estamos convencidos que es positivo que exploren y reflexionen sobre el problema, resolviendo cada uno, al menos esta instancia pequeña.

Al analizar las palabras utilizadas, al momento de responder sobre la estrategia utilizada en el modo de videojuego libre final, observamos que surgen palabras que antes no habían sido utilizadas: *estrategia*, *algoritmo*, *sort*. Así, notamos la aparición de términos que, si bien no les eran ajenos al inicio de la clase, recién les surge mencionarlos al terminarla. Esto evidencia una apropiación de los términos y vocabulario específico de la temática.

Una limitación que tuvimos, y que no fue contemplada, fue que todos los modos del videojuego requieren del uso del mouse ya que las acciones se realizan haciendo click. Por lo tanto, no hemos podido capturar los juegos de los estudiantes que participaron de las clases desde otros dispositivos, como ser tableta o celular.

Por último, creemos que las herramientas virtuales tienen un gran potencial para hacer seguimiento de los estudiantes, con gran detalle, no sólo de los que en la modalidad presencial hubiesen participado de manera activa sino también de los que lo hubieran hecho de manera pasiva. Es una gran oportunidad para desarrollar y mejorar estrategias de docencia, detección y de intervención, para estudiantes que requieren de asistencia adicional. Sin embargo, es esencial no

perder de vista que un intercambio personal, entre estudiantes y docentes, puede proporcionar toda esta información e incluso mucha más.

A modo de cierre nos llevamos como asignatura pendiente pensar, para el momento del retorno a la presencialidad, cómo incorporar los recursos y estrategias que fueron efectivos en la virtualidad.

Referencias

1. Broda, K., Khoshnevisan, H., Eisenbach, S.: Reasoned Programming. Prentice Hall (1994)
2. Charlot, B.: La epistemología implícita en las prácticas de enseñanza de las matemáticas. In: Conferencia dictada en Cannes (1986)
3. Communications, N.: Javascript (1995), <https://web.archive.org/web/20070916144913/http://wp.netscape.com/newsref/pr/newsrelease67.html>
4. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to algorithms. MIT press (2009)
5. Dougiamas, M.: Moodle (2002), <https://moodle.org/>
6. Ellson, J., Gansner, E., Koutsofios, L., North, S.C., Woodhull, G.: Graphviz—Open Source Graph Drawing Tools. In: Mutzel, P., Jünger, M., Leipert, S. (eds.) Graph Drawing. pp. 483–484. Springer Berlin Heidelberg, Berlin, Heidelberg (2002)
7. Gries, D.: The science of programming. Springer Science & Business Media (2012)
8. gskinner: Createjs (2018), <https://createjs.com/>
9. Inc., M.: Mongodb (2009), <https://www.mongodb.com/>
10. Koedinger, K.R., Kim, J., Jia, J.Z., McLaughlin, E.A., Bier, N.L.: Learning is not a spectator sport: Doing is better than watching for learning from a mooc. In: Proceedings of the second (2015) ACM conference on learning@ scale. pp. 111–120 (2015)
11. Lee, S.J., Jung, A., Park, J., Yun, M.: Cost-efficient hands-on learning design for computer organization course. In: 2020 15th International Conference on Computer Science & Education (ICCSE). pp. 150–155. IEEE (2020)
12. Loper, E., Bird, S.: Nltk: The natural language toolkit. In: In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics (2002)
13. Lopez-Rosenfeld, M.: ”tell me and i forget, teach me and i may remember, involve me and i learn”: changing the approach of teaching computer organization. In: 2017 IEEE/ACM 1st International Workshop on Software Engineering Curricula for Millennials (SECM). pp. 68–71. IEEE (2017)
14. Mueller, A.: amueller/word_cloud, https://github.com/amueller/word_cloud
15. Reichle, E.D., Pollatsek, A., Fisher, D.L., Rayner, K.: Toward a model of eye movement control in reading. Psychological review **105**(1), 125 (1998)
16. Ronacher, A.: Flask (2010), <https://flask.palletsprojects.com/en/2.0.x/>
17. Van Rossum, G., Drake, F.L.: Python 3 Reference Manual. CreateSpace, Scotts Valley, CA (2009)
18. Waskom, M.L.: seaborn: statistical data visualization. Journal of Open Source Software **6**(60), 3021 (2021). <https://doi.org/10.21105/joss.03021>, <https://doi.org/10.21105/joss.03021>