09-024

Custom PERT problems to encourage individual original work

Antonio Arauzo-Azofra; Lorenzo Salas-Morera; Elena Sánchez-López; Laura García-Hernández; Juan María Palomo-Romero

Universidad de Córdoba;

Despite the project manager (PM) usually ends up using computer tools for project scheduling, generally, it is recognized that knowing how to build activity networks and calculate its times is indispensable for their training. To understand these concepts, an explanation is not enough. It is paramount to face a problem and solve it personally. However, when a problem is the same for the whole class, it is common to try to copy or learn from someone else's solution. In this paper, we present the results of an experience with several groups of students who either had a list of identical problems or a list of personalized problems. When using personalized problems for each student, we have found that there is an incremental positive impact on the development of project management competencies at the end of the course.

Keywords: PERT; Problem solving; Project scheduling

Problemas PERT personalizados para incentivar el trabajo individual original

Aunque el director de proyectos termina utilizando herramientas informáticas para la programación de proyectos, en general, se reconoce como necesidad que conozca cómo se construyen las redes de actividades y cómo se calculan sus tiempos. Para comprender estos conceptos, no basta con una explicación, es fundamental enfrentarse al problema y resolverlo personalmente. Sin embargo, cuando se da una relación de problemas iguales a una clase, es habitual que surja la tentación de copia o simplemente de intentar aprender de la solución de otro. En este trabajo, exponemos los resultados de una experiencia con varios grupos de estudiantes, unos con relaciones de problemas iguales y otros con relaciones de problemas diferentes a cada estudiante. Verificamos que hay un impacto positivo en el desarrollo de estas competencias al final de curso, al usar problemas personalizados para cada estudiante.

Palabras clave: *PERT; Resolución de problemas; Programación de Proyectos*

Correspondencia: Antonio Arauzo arauzo(at)uco.es

Agradecimientos: Agradecemos a Jordi Bordoy Nieto su colaboración en el desarrollo del programa generador de problemas.

1. Introducción

La asignatura de Proyectos de cuarto curso del grado de Ingeniería Informática comprende una serie de competencias necesarias para la organización y gestión de Proyectos, tales como capacidad de resolución de problemas, capacidad de análisis y síntesis, capacidad de gestión de la información, o capacidad de liderazgo, entre otras. Igualmente, el programa de la asignatura comprende los conocimientos básicos de gestión de proyectos y, entre ellos, los relacionados con el método PERT/CPM. En este contexto, resulta de vital importancia que los estudiantes demuestren su capacidad de trabajo autónomo resolviendo un conjunto de problemas de clase, enfrentándose con diversas formas de analizar la información y resolver problemas (problemas de clase, ejercicios individuales, prácticas con programas informáticos, etc.), lo que sin duda contribuirá a enriquecer su aprendizaje y a fortalecer su capacidad de resolución de problemas (Miskioğlu, 2016) . Para ello tradicionalmente se ha asignado una tarea consistente en la resolución individual de un paquete de problemas de dificultad creciente, con el objeto de que el estudiante pueda ejercitar sus conocimientos, relacionar la teoría con la práctica y adquirir suficiente práctica de cara al examen (Green & Jax, 2011), pero esto presenta la dificultad de la necesidad de generar anualmente un conjunto de problemas suficientemente extenso. Por otro lado, si la relación de problemas de que se dispone no es individualizada, se corre el riesgo de que los estudiantes, simplemente copien el resultado de otros compañeros, o incluso se distribuyan el trabajo entre varios, con lo que la consecución del objetivo general de la tarea puede quedar muy comprometido.

Para soslayar los problemas mencionados, puede ser de utilidad disponer de una herramienta que genere un número suficientemente grande de problemas de forma automática, especialmente cuando el número de estudiantes es elevado (Poch, Boada, Soler, & Prados, 2016). Esto evita tener que recurrir a colecciones de problemas obtenidos de la bibliografía, cursos, o páginas web educativas y además tiene la ventaja de que se evita el plagio de unos estudiantes a otros generando colecciones de problemas individualizadas (Ahmed, Gulwani, Redmond, & Karkare, 2013). Además, antes de adjudicar las colecciones de problemas es necesario asegurarse de que la dificultad es: a) creciente de unos problemas a otros y b) similar de unas colecciones a otras (Singh, Gulwani, & Rajamani, 2012).

Los grafos utilizados en el método PERT/CPM o el método de la ruta crítica (CPM), que están dentro de las técnicas mencionadas en el PMBoK para realizar la programación de un proyecto, utilizan grafos tipo AOA (*Activity On Arc*). En ellos, se representa la programación de un proyecto con los arcos como tareas y los nodos como sucesos del proyecto.

Los grafos AOA presentan una estructura generalmente más sencilla y ordenada del proyecto que los grafos tipo AON (*Activity On Node*), porque organizan prelaciones comunes a varias actividades en sucesos. Sin embargo, necesitan la inclusión de actividades ficticias para crear esta organización y, por tanto, su construcción es más compleja (Krishnamoorthy, 1979). No obstante, es perfectamente abordable su desarrollo a mano para problemas pequeños y nos parece muy recomendable aprender a construirlos para comprender mejor las estructuras e interacciones que se dan entre las actividades de los proyectos.

2. Objetivos

- Aumentar la motivación y el rendimiento de los estudiantes, para ello, se propone la utilización y se describe el diseño de un generador de problemas del método PERT/CPM que incluyen la construcción de un grafo AOA y el calculo de tiempos sobre el mismo.

- Evaluar el uso de baterías de problemas generados diferentes como ejercicios para un curso, comparándolo con la utilización de relaciones de problemas idénticos para cada estudiante.

3. Metodología

En esta sección, describimos en primer lugar el diseño del generador de problemas que incluye la generación, la resolución y la representación de los mismos. A continuación, explicamos la experiencia docente, describiendo los estudiantes objetivo, la configuración de las baterías de problemas y la preparación de las encuestas de valoración.

3.1 Generador de problemas

El generador de problemas es una aplicación web que permite ver el enunciado del problema, unas pistas para su resolución y la solución. La generación de la tabla de actividades y sus prelaciones se hace con el algoritmo que describimos a continuación. Éste garantiza que las prelaciones no formen ciclos (A → B → A), garantiza que no haya prelaciones redundantes (A → B, B → C, C → D, B → D) y permite regular el tamaño y la complejidad de los problemas.

Los parámetros del algoritmo, que definirán las propiedades del grafo que dará solución al problema generado son:

- *N* – Número de actividades que tendrá el grafo ([1, +∞[ )
- *MS* – Máximo Siguientes ([1, +∞[ ): número máximo de actividades siguientes que tendrá cada actividad del grafo. Esto nos permite limitar la complejidad del grafo generado respecto a su tamaño.
- *PS* – Proximidad Siguientes ([1, +∞[ ): número de actividades entre las que se elegirán las siguientes. Esto nos permite limitar la complejidad del grafo generado respecto a su dificultad de resolución. De esta forma, para un mismo tamaño de grafo en número de actividades y número de prelaciones, tendremos un grafo con menos relaciones entremezcladas si limitamos las siguientes a ser escogidas entre un grupo de actividades próximas más reducido.

El algoritmo desarrollado se muestra en la Tabla 1. En su paso 4, recorremos la lista en orden porque esto permite que el grafo tenga prelaciones más ordenadas y, por tanto, más claras para resolver el problema. Consideramos que la actividad de ordenarlas o trabajar contra un orden alfabético, aunque pueda ser un buen ejercicio mental, no es productiva para el aprendizaje de los esquemas de resolución que buscamos. Es fácil ver que esto no afecta a la forma de los problemas, porque el efecto de recorrerlas en otro orden sería el mismo que el de barajar las actividades y siempre daría lugar a generar un problema estructuralmente equivalente.

La garantía de que no se generan ciclos en las prelaciones de las actividades se consigue incluyendo sólo actividades que no han sido procesadas todavía en las actividades candidatas a siguientes. Por otra parte, se garantiza que no aparecerán prelaciones redundantes retirando las actividades inmediatamente siguientes de todas las precedentes a la actividad actual de las candidatas.

A parte de generar problemas que sean adecuados para su resolución, también es necesario generar sus respectivas soluciones. Éstas podrán usarse para la corrección o como referencia para que los estudiantes comprueben si su solución es correcta, por ejemplo en una batería de ejercicios para practicar. También se pueden utilizar para mostrar

Tabla 1.- Algoritmo generador de una tabla de actividades con sus predecesoras

1. *TablaPredecesoras* = lista de listas vacía (con una lista vacía para cada actividad)
2. *PrecedentesR* = lista de listas vacía (idém, almacenará para cada actividad una lista de todas sus precedentes, incluye las predecesoras de las predecesoras recursivamente)
3. *Actividades* = lista de actividades, cuyos nombres se generan escogiendo los *N* primeros elementos de la secuencia infinita {A, B, C, … , Z, AA, AB, … , ZZ, AAA, …}.
4. Para *i* desde 1 hasta *N:* (tal que $actividad_i \in Actividades$)

   Para cada *predecesora* en *TablaPredecesoras*[*actividad*]:

   $PrecedentesR[actividad_i]$ += {*predecesora*} U $PrecedentesR[predecesora]$

   $Candidatas = \{ actividad_k \mid i < k \leq i + PS \} - Siguientes(PrecedentesR[actividad_i])$

   Num_siguientes = número aleatorio con distribución uniforme entre 1 y *MS*

   Para cada *siguiente* en Muestra(*Candidatas*, *Num_siguientes)*:

   $TablaPrecedecesoras[siguiente] += \{ actividad_i \}$

5. Devuelve *TablaPredecesaras*

donde: $Siguientes(A) = \{ s \mid \exists a \in A : s \in TablaPredecesoras[a] \}$, y

Muestra(*C*, *N*) escoge aleatoriamente con probabilidad uniforme *N* elementos del conjunto *C*

parcialmente algunas características de la solución. De esta forma, se permite al estudiante comprobar si lo ha hecho bien pero obligándole a hacerlo porque no tiene la solución. Esta última opción es la que hemos tomado para nuestra batería de problemas. La tabla 2 muestra un ejemplo completo de un problema sencillo generado.

A la hora de representar la solución, hay que tener en cuenta que la construcción de un grafo PERT no es un problema de solución única. Se necesita la inclusión de actividades ficticias y éstas pueden incluirse de diversas formas dando varias estructuras que son correctas —dado que incluyen todas las prelaciones del problema y no incluyen ninguna prelación no definida en él. No obstante, cuanto menor sea el número de actividades ficticias, el grafo será más sencillo, la representación del proyecto será más clara y el tiempo a emplear en fases posteriores será menor. Esto limita las soluciones deseables a un conjunto mucho más reducido.

Tabla 2.- Ejemplo de problema generado

Dada la siguiente tabla de prelaciones, se pide: a) construir su grafo PERT, b) calcular sus tiempos *early*, c) calcular su duración, y d) calcular sus tiempos *last.*

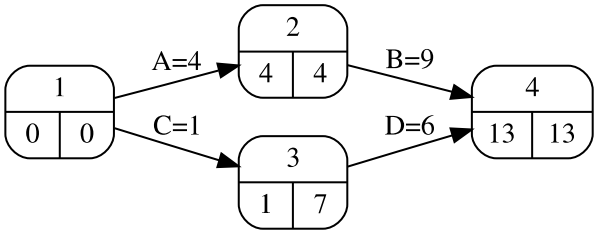
Actividad	Duración	Precede a
A	4	B
B	9	
C	1	D
D	6	

Pista: Se puede construir un grafo con 4 nodos, 0 actividades ficticias y 2 caminos.

Elegimos el algoritmo de Gento-Municio (2004) porque en un tiempo razonable es el que genera grafos con el menor número de actividades ficticias de los comparados en este trabajo (Arauzo-Azofra, Salas-Morera, Garcia-Hernandez, & Perez-Caballero, 2015) Está implementado en el lenguaje de programación *Python* e incluido en el software *PPC-project* (Salas, 2013). El repositorio del proyecto incluye su código fuente, para poder ser estudiado y extendido por otros investigadores.

En la Figura 1 se puede ver como se representa la solución del problema generado de la Tabla 2.

Figura 1: Ejemplo de grafo PERT con la solución del problema de la Tabla 2



3.2 Participantes y procedimiento

El experimento se ha llevado a cabo en la asignatura de Proyectos, la cual tiene carácter obligatorio y corresponde al cuarto curso de los distintos Grados impartidos en la Escuela Politécnica Superior (EPSC) de la Universidad de Córdoba.

El grupo objeto de estudio, pertenece al Grado de Ingeniería Informática (GIINF). Está formado por 39 alumnos y ha sido dividido en dos subgrupos: el primero, en adelante denominado *G.Variados,* con 20 participantes y el segundo, denominado *G.Iguales,* con 19 participantes. A todos los alumnos del grupo *G.Variados* se les ha hecho entrega de una relación de problemas variados, es decir, diferente a la de sus compañeros, mientras que los problemas de *G.Iguales* fueron iguales para todos sus integrantes. Las entregas han sido creadas con el generador de problemas anteriormente descrito y cada una cuenta con cinco ejercicios de dificultad creciente en su resolución. En cada uno de ellos se pide; construir el grafo PERT, calcular los tiempos *early*, los tiempos *last* y la duración del proyecto.

En la Tabla 3, se pueden observar, para las entregas de problemas variados, el número de actividades que componen cada ejercicio, así como los intervalos en los que pueden oscilar el número nodos, actividades ficticias y posibles caminos, necesarios para llegar a la solución.

Tabla 3.- Características de las entregas de problemas variados

Ejercicios iguales				
	Actividades	Nodos	Ficticias	Caminos
Ejercicio 1	4	4-5	0-3	2- 4
Ejercicio 2	7	6-8	0-4	3-6
Ejercicio 3	12	9-14	1-11	7-32
Ejercicio 4	20	15-27	5-21	
Ejercicio 5	26	18-31	7-29	

Los ejercicios de las entregas iguales para todos los integrantes del grupo, presentan el mismo número de actividades que los ejercicios variados. Sus características se muestran en la Tabla 4.

Tabla 4.- Características de las entregas de problemas iguales

Ejercicios iguales				
	Actividades	Nodos	Ficticias	Caminos
Ejercicio 1	4	4	0	2
Ejercicio 2	7	6	1	5
Ejercicio 3	12	9	4	13
Ejercicio 4	20	17	10	31
Ejercicio 5	26	26	20	37

Todos los alumnos han trabajado en los ejercicios en horario fuera de clase, aplicando los conocimientos adquiridos desde el inicio del curso. El periodo establecido para la resolución ha sido de tres semanas, un tiempo prudente, dado que los problemas 4 y 5 presentaban una dificultad considerable. Una vez cumplido este plazo, y teniendo en cuenta que todos los alumnos realizaron la entrega dentro los límites, se ha procedido a su corrección, evaluándose la totalidad de los ejercicios sobre 10 puntos.

Así, el objetivo de esta experiencia es analizar cómo influye cada caso en la capacidad de los alumnos para resolver los ejercicios en el examen final.

4. Resultados

Las actividades evaluables en torno a la competencia “capacidad para resolver problemas” y concretamente, problemas de programación de proyectos, han sido tres: entrega de la colección de problemas, prácticas con software de proyectos y una parte del examen final. Se plantea la hipótesis de que los estudiantes que realizan la colección de problemas variados (*G.Variados*) desarrollan mejor su capacidad de resolución de problemas y, por lo tanto, obtienen mejores calificaciones en la parte correspondiente del examen final. En la tabla 5 se presentan los resultados globales de cada uno de los subgrupos para cada una de las actividades desarrolladas.

Los resultados obtenidos por cada uno de los subgrupos parecen prometedores en el sentido de que los estudiantes que realizaron la colección de problemas individualizada (*G.Variados*) obtuvieron por término medio mejor calificación que los del grupo que realizó la misma colección de problemas (G.Iguales).

Sometida esta hipótesis a un test t de Student de diferencia de medias, se obtienen los resultados que aparecen en las tablas 6, 7, y 8. A la vista de los resultados, midiendo el rendimiento con los resultados del examen, puede concluirse que, en efecto, la resolución de ejercicios variados, individualizados y con dificultad creciente ha contribuido a mejorar la

Tabla 5.- Resumen de las calificaciones de los subgrupos para cada actividad.

	Prácticas con software de proyectos		Colección de problemas		Examen	
	G.Variados	G.Iguales	G.Variados	G.Iguales	G.Variados	G.Iguales
n	20	19	20	19	20	19
Calificación media	**9.1**	8.8	**7.9**	7.5	**8.6**	7.4
Desv. típica	0.5	1.0	2.5	2.5	2.0	1.9

Tabla 6.- Test t de Student para las diferencias de medias en las calificaciones de prácticas.

	G.Variados	G.Iguales
Media	9,1	8,8
Varianza	0,25	1,06
Observaciones	20	19
Diferencia media observada	0,3421	
Varianza de las diferencias	1,2237	
Grados de libertad	18	
Estadístico t	1,3480	
P (T<=t) unilateral	0,0972	
t crítica unilateral (α = 0.05)	1,7341	

Tabla 7.- Test t de Student para las diferencias de medias en las calificaciones de los ejercicios.

	G.Variados	G.Iguales
Media	7,9	7,5
Varianza	6,04	6,43
Observaciones	20	19
Diferencia media observada	0,6368	
Varianza de las diferencias	10,1124	
Grados de libertad	18	
Estadístico t	0,8729	
P (T<=t) unilateral	0,1970	
t crítica unilateral (α = 0.05)	1,7341	

Tabla 8.- Test t de Student para las diferencias de medias en las calificaciones del examen.

	G.Variados	G.Iguales
Media	8,6	7,4
Varianza	3,96	3,63
Observaciones	20	19
Diferencia media observada	1,1090	
Varianza de las diferencias	6,7304	
Grados de libertad	18	
Estadístico t	1,8634	
P (T<=t) unilateral	0,0394	
t crítica unilateral (α = 0.05)	1,7341	

capacidad de resolución de problemas de los estudiantes, con respecto a los que realizaron la misma colección para todo el grupo.

En las prácticas realizadas simultáneamente con software de gestión de proyectos, aunque aparentemente la media sea mejor, la diferencia no es significativa. Es posible que sea porque las competencias evaluadas en este tipo de actividad no estén tan directamente relacionadas con las desarrolladas con los ejercicios.

El hecho de que en la evaluación de los problemas la diferencia no sea significativa muestra que en el *G.Iguales* han cumplido entregando los problemas correctamente resueltos pero quizá se algunos se hayan basado en algunas soluciones de otros compañeros que tenían problemas idénticos. Esto no es posible detectarlo en la evaluación de los ejercicios. Por tanto, el hecho de utilizar problemas diferentes ha obligado a los miembros del G.Variados a desarrollar la competencia de resolver los problemas mejor.

Para tener una idea más clara y mayor información sobre el funcionamiento de la experiencia, pasamos una encuesta a los estudiantes participantes con las preguntas que se muestran en la Tabla 9. La encuesta ha sido anónima. Aunque esto nos impide poder relacionar uno a uno sus respuestas con los resultados, creemos que es más importante el anonimato para tener unas respuestas más sinceras y motivar la participación. Aun así, 3 estudiantes de cada grupo han dejado la encuesta sin responder. De las encuestas podemos extraer las siguientes conclusiones:

- Todos valoran muy positivamente haber hecho la relación de problemas para preparar el examen. La diferencia entre los dos grupos es muy pequeña.
- En el grupo *G.Variados* tenían claro que sus problemas eran distintos a los de sus compañeros. Sin embargo, al estar mezclados en clase, una mayoría de los estudiantes del grupo *G.Iguales* pensaron que sus problemas también eran diferentes Sería interesante comparar en el futuro con un grupo donde toda la clase tenga la misma relación de problemas.
- La dificultad de los problemas la han valorado como adecuada en ambos grupos.
- Aunque no hay gran diferencia, en el grupo *G.Iguales* más estudiantes han resuelto los problemas junto con otros compañeros.

Tabla 9.- Resumen de las encuestas sobre la experiencia.

Preguntas	G.Variados	G.Iguales
¿Te ha ayudado hacer la relación de problemas PERT para preparar esa parte del examen? (1: poco .. 5: mucho)	4.35	4.44
¿Te diste cuenta de que tus problemas eran diferentes(G.Variados) / iguales(G.Iguales) a los de otros compañeros?	94%	38%
Los problemas te han resultado… (1: muy fáciles .. 5: muy difíciles)	3.00	3.31
¿Te juntaste con otros compañeros para trabajar en los problemas y ayudaros con las dudas?	41%	56%
Dirías que resolviste los problemas… (1: sin mirar otras soluciones .. 5: basándome en otras)	2.29	1.94
n. respuestas	17	16

- En ambos grupos indican que los problemas los han resuelto mirando poco otras soluciones. Admiten haber mirado algo más otras soluciones en el grupo *G.Variados*. Sin embargo, deben referirse a soluciones a problemas similares porque nadie tenia acceso a la solución de sus problemas y en el grupo *G.Variados* eran únicos.

5. Conclusiones

Este artículo presenta el diseño de una herramienta para la generación automática de problemas del método PERT/CPM y sus respectivas soluciones. Esta herramienta es capaz de generar problemas de manera rápida, permitiendo, además, configurar su complejidad de resolución con tres variables.

Para demostrar la idoneidad de esta herramienta para el apoyo de la formación universitaria en gestión de proyectos, se ha llevado a cabo una experimentación en la asignatura de Proyectos, correspondiente Grado de Ingeniería Informática impartido en la Escuela Politécnica Superior de la Universidad de Córdoba.

Los resultados del experimento, en torno a la competencia evaluable de “capacidad para resolver problemas”, han sido comparados mediante tres calificaciones: prácticas, colección de problemas y examen final de la asignatura. En término medio, cabe destacar que para las tres pruebas, el grupo que ha realizado la colección de ejercicios variados (*G.Variados*) ha obtenido mejores calificaciones. La diferencia se hace notable, especialmente, en la calificación del examen final de la asignatura, en la que este grupo de alumnos ha conseguido una nota media 1.2 puntos superior al del otro grupo de alumnos. El test t de Student confirma que existen diferencias significativas en las calificaciones del examen final entre ambos grupos. De esta manera, queda demostrado mediante las evaluaciones a los propios alumnos, que la adquisición de conocimientos mediante la estrategia de generación de problemas variados con esta nueva herramienta se produce de una manera más natural y fluida que por el método tradicionalmente utilizado.

Entre las futuras mejoras que podrían ser implementadas en esta herramienta, cabría destacar la inclusión de estimaciones de tiempos (optimista, más probable y pesimista), el cálculo de holguras, la planificación del proyecto mediante el diagrama de Gantt y el cálculo de tiempos ROY (AON). Sería interesante complementar la evaluación comparando los resultados con grupos de clases separadas para evitar la interferencia detectada al pensar los estudiantes del grupo *G.Iguales* que sus problemas también eran distintos. Finalmente, nos gustaría que otros profesores usasen la herramienta y validar con ellos estos resultados.

Bibliografía

Ahmed, U. Z., Gulwani, S., Redmond, M., & Karkare, A. (2013). Automatically Generating Problems and Solutions for Natural Deduction. In *23rd International Joint Conference on Artificial Intelligence, IJCAI* (pp. 1968–1975). Beijing.

Arauzo-Azofra, A., Salas-Morera, L., García-Hernández, L., Perez-Caballero, A. (2015). A comparison of algorithms for constructing PERT graphs for large projects. In

Gento-Municio, A. M. (2004). Un algoritmo para la realización de grafos con las actividades en los arcos, grafos PERT. *Cuadernos Del CIMBAGE*, (7), 103.

Green, K., & Jax, C. (2011). Problem solvers are better leaders: facilitating critical thinking among educators through online education. *Procedia - Social and Behavioral Sciences*, *15*, 727–730. https://doi.org/10.1016/j.sbspro.2011.03.173

Krishnamoorthy, M. S., & Deo, N. (1979). Complexity of the minimum-dummy-activities problem in a pert network. *Networks*, 9(3), 189–194. http://doi.org/10.1002/net.3230090302

Miskioğlu, E. E. (2016). Developing More Robust Problem Solvers through Diversity of Course Experiences. In *Frontiers in Education Conference*. https://doi.org/10.1109/FIE.2016.7757592

Poch, J., Boada, I., Soler, J., & Prados, F. (2016). Automatic creation and correction of mathematical problems. *International Journal of Engineering Education*, *32*(1), 150–162.

Salas-Morera, L., Arauzo-Azofra, A., García-Hernández, L., Palomo-Romero, J. M., & Hervás-Martínez, C. (2013). PpcProject: An educational tool for software project management. *Computers & Education*, *69*, 181-188.

Singh, R., Gulwani, S., & Rajamani, S. (2012). Automatically Generating Algebra Problems. *AAAI Conference on Artificial Intelligence*, 1620–1627.