

APRENDIZAJE BASADO EN RESOLUCION DE PROBLEMAS EN INGENIERIA INFORMATICA. CASO PRACTICO: RESOLVEDOR AUTOMATICO DE SUDOKUS

Pablo Sáez
Departamento de Ingeniería Informática
Universidad de Concepción
Casilla 160-C, Concepción Correo 3
psaez@inf.udec.cl

César Monsalve
Departamento de Ingeniería Informática
Universidad de Concepción
Casilla 160-C, Concepción Correo 3
cemonsal@udec.cl

Area temática: innovaciones metodológicas en el proceso aprendizaje-enseñanza.

Resumen: Exponemos brevemente el sistema de aprendizaje basado en resolución de problemas, que es la base de los currículums de un buen número de universidades en Canadá, Australia, Dinamarca, México y otros países. La idea es enfrentar al alumno a problemas concretos en un ramo, en formato de “proyecto”, dejando la “asimilación de información” más bien a cargo del propio alumno, con libros. El profesor actúa como un asesor, un orientador más que como un “transmisor de información”. Comentamos algunas posibilidades de implementar este sistema en la práctica sin irse al extremo tan radical planteado en la propuesta original, el cual puede, en nuestra opinión, fallar en culturas como la nuestra. Mostramos un ejemplo práctico de la aplicación de estas ideas: el desarrollo por parte de los alumnos de un resolvedor automático de sudokus como una forma de estudiar los conceptos relacionados con la complejidad computacional y la NP-completitud en la ciencia de la computación.

1. INTRODUCCION

Nos interesa en este artículo exponer brevemente el método de aprendizaje basado en resolución de problemas (sección 2), que consiste esencialmente en que el alumno se ve enfrentado a problemas concretos, prácticos, en el transcurso de las asignaturas, asumiendo el profesor un rol de asesor, o de guía más que el de “transmisor de información”. La idea es un poco la del proverbio chino que dice “lo que escucho, lo olvido; lo que veo, lo recuerdo; pero lo que hago, lo aprendo”.

La solución en su forma comúnmente adoptada es un tanto radical en nuestra opinión. El alumno debe por cuenta propia leerse los contenidos en los libros correspondientes, consultando al profesor las dudas. Este enfoque es desilusionante para algunos, pero por sobre todo nos parece que en culturas como la nuestra el sistema puede tender a “chacarse” como comúnmente se dice, en la medida en que se presupone que el alumno va a, disciplinadamente, ocupar las tardes que tiene asignadas para estudiar en eso (estudiar).

Proponemos buscar un enfoque intermedio, que combine las ventajas de enfrentar al alumno a problemas concretos en el transcurso de la asignatura, sin que necesariamente disponga de una total libertad en el uso del tiempo. Mostramos un ejemplo concreto de estas ideas: la construcción por parte de los alumnos de un resolvidor automático de sudokus como ejemplo práctico en el estudio de los temas de complejidad computacional y NP completitud en la ciencia de la computación.

2. APRENDIZAJE BASADO EN RESOLUCION DE PROBLEMAS

Tal vez sea una buena idea partir con una metáfora proveniente de la biología: algunas especies de hormigas usan una técnica conocida como “andar en tándem” para mostrarle a otra hormiga cómo ir desde el nido a una fuente de comida [1]. La hormiga “profesora” guía a la “alumna”, llevándola y corrigiéndole el rumbo. En contraste, todos conocemos el comportamiento de la abeja, que cuando sabe de la ubicación de algún alimento llega a la colmena y realiza una danza a la cual “asisten” las otras abejas, pasivamente por así decirlo. Hay en este caso tan sólo una “transmisión de información”. Y la pregunta que surge entonces para uno mismo como docente es: ¿estoy siendo “abeja” u “hormiga”?, es decir: ¿me limito a realizar una “danza” adelante, en el pizarrón, pretendiendo “transmitir información” a los alumnos, o estoy realmente involucrado en su proceso de aprendizaje y trato de guiarlos en su actuar, corrigiéndolos y orientándolos, como la hormiga?

No es difícil darse cuenta de que en la academia la gran mayoría de los profesores somos “abejas”, sumergidos en un afán por “transmitir conocimiento”, considerado éste como un cúmulo de datos. Sin embargo al momento de enseñar ciencia, o ingeniería, lo que debería propiciarse es más bien el razonamiento autónomo en los alumnos, la habilidad de reflexionar en forma crítica y el autoaprendizaje; pero la pregunta evidente es: ¿cómo?

Existen distintos modelos educativos: aquellos basados en escuchar y repetir, aquellos basados en observar y reproducir, y aquellos basados en percibir y comprender. El modelo al cual queremos referirnos aquí, brevemente, por las limitaciones del espacio, es el modelo PBL [2,3], por su sigla en inglés: Problem solving Based Learning (aprendizaje basado en resolución de problemas), en donde la idea central es enfrentar a los alumnos a problemas

concretos, prácticos, en el transcurso de un ramo. El ramo pasa a ser una especie de proyecto durante el cual el profesor hace de guía para el equipo de trabajo, ayudándolos u orientándolos. Desde luego hay un proceso de estudio, de “asimilación de contenidos”, pero de esta parte del proceso de aprendizaje es responsable el propio alumno, quien debe leerse los capítulos correspondientes del libro correspondiente en el transcurso del ramo, estando el profesor evidentemente encargado de aclarar las dudas que se le presentan al alumno en su lectura del texto; pero ya no encargado de “danzar” en el pizarrón, transmitiendo información. Porque no es difícil darse cuenta de que si el alumno pierde el hilo de esta “transmisión de información” van a ser vanos los esfuerzos por pretender “transmitirle más información” a un receptor que ya no está procesando. En contraste con esta situación, reuniones de trabajo o conversaciones grupales siempre son más eficientes al momento de entender conceptos o relacionarlos.

Desde luego este tipo de modelos de aprendizaje no es bienvenido por todos los alumnos. Supimos de un caso en el cual una alumna estaba bastante molesta porque consideraba que era una estafa pagar tal cantidad de dinero para que simplemente la mandaran a la biblioteca a leer. Estadísticamente un 25% de los alumnos aproximadamente prefiere en realidad el sistema tradicional, de “entrega de información”. Un 25% prefiere PBL y un 50% dice que puede aprender indistintamente con uno u otro método. Hay por lo tanto un grado de resistencia al cambio al momento de introducir este tipo de metodologías, lo cual es natural por lo demás, esperable. La distribución semanal de las actividades por ejemplo es muy distinta a la usual. El trabajo en proyectos, que es lo que finalmente hace el alumno, es muy demandante. Y se le asigna bastante tiempo libre al alumno, de modo que disponga del tiempo necesario para leerse los capítulos de los libros que corresponde. Típicamente un alumno cursa solamente *dos* ramos al mismo tiempo, los ramos son de duración variable etc.

Digamos, para terminar esta sección, que el modelo PBL se ha aplicado con éxito en distintos lugares del mundo, incluyendo Canadá, Australia, Dinamarca, México etc.

3. NUESTRA PEQUEÑA EXPERIENCIA

Nuestro punto de vista personal en esta temática es desde luego favorable a este modelo, pero al mismo tiempo realista: en PBL un alumno puede tener por decir algo un día a la semana entero más tres tardes enteras libres, de modo tal de disponer del tiempo necesario para estudiar en la biblioteca. Es decir que se presupone un cierto grado de disciplina y responsabilidad del alumnado que en nuestra cultura tiende a faltar. Además evidentemente no hay recetas en esto de los modelos de aprendizaje, sino más bien propuestas o enfoques. Cada establecimiento es desde luego dueño de adoptar los métodos o sistemas que estime conveniente.

Nuestra percepción de la situación es que puede resultar adecuado un enfoque intermedio, mixto: es decir que sin irse al extremo radical de cambiar completamente los métodos de enseñanza, las mallas curriculares y el sistema en general, uno puede como docente encauzar el desarrollo de las asignaturas por caminos de corte más práctico, interactuar con los alumnos, planteándoles problemas a resolver, corriendo el riesgo de “perder” una o dos horas de clase completas con los alumnos estudiando algún problema a veces sencillo, manteniéndose uno como docente en un papel de asesor del equipo de trabajo. Y hay una ventaja extra de este tipo de actividades, que es que el docente se entera de si los alumnos están captando *algo* de lo que se está viendo en clases. De lo contrario, al “actuar como abeja”, es probable que la hora o el par de horas hayan sido de todos modos tiempo perdido.

En este sentido nos hemos preocupado, en la medida de nuestras posibilidades, de enfatizar el trabajo práctico de los alumnos, en la línea de las ideas expuestas más arriba. Ahora, ciertamente que el alumno siempre resuelve problemas de carácter práctico en los distintos ramos, por ejemplo al momento de rendir certámenes o tests, pero el sentimiento tiende a ser más bien negativo, es decir “me fue más o menos mal (o más o menos bien), hice lo que pude en tal o cual problema” etc. Quisiéramos exponer entonces aquí brevemente un ejemplo de actividad práctica por parte de alumnos: la construcción de un resolvidor automático de sudokus como ejemplo en el estudio de la teoría de la NP-completitud en ciencia de la computación.

4. EL EJEMPLO

Sin entrar en detalles técnicos excesivos, digamos que el problema de Satisfacibilidad de Fórmulas Proposicionales Booleanas (SAT) fue el primer problema identificado como perteneciente a la clase de complejidad NP-completo [4], es decir que hay una clase importante de problemas (la clase NP) que puede ser “eficientemente traducida” a una instancia de SAT, es decir a una fórmula proposicional que resuelve el problema original si es que se puede encontrar un conjunto de valores de verdad para las proposiciones que haga verdadera a la fórmula. Se podría decir que los programas que resuelven SAT constituyen una tecnología de razonamiento automático, que ha encontrado muchas aplicaciones a nivel industrial en las últimas décadas en variadas áreas tales como verificación de diseño de circuitos digitales, configuración de horarios e inteligencia artificial por nombrar algunos. Los problemas SAT son resueltos por un algoritmo comúnmente llamado solver, el cual automáticamente busca la solución al problema planteado. El sistema de resolución de un SAT-solver se podría ver como una caja negra, en la cual no se necesita una mayor interacción entre el usuario y el solver.

La entrada al solver, o input, es una fórmula proposicional booleana expresada en Forma Normal Conjuntiva (CNF), la cual es una conjunción (Y lógico (&)) de una o más cláusulas. Una cláusula es una disyunción (O lógico (V)) de uno o más literales. Por su parte, un literal es una instancia positiva o negativa de una letra proposicional. Por ejemplo, las siguientes formulas están en CNF:

$$\begin{aligned} & A \& B \\ & \neg A \& (B \vee C) \\ (A \vee B) \& (\neg B \vee C \vee \neg D) \& (D \vee \neg E) \end{aligned}$$

Como output el solver responde sí o no, dependiendo de la satisfacibilidad de la formula. Si la instancia es satisfacible, algunos SAT-solver pueden entregar una valuación o modelo que hacen que la fórmula ingresada como input sea verdadera. Para modelar el problema, el principal desafío es expresar todas las restricciones del problema original en fórmulas proposicionales booleanas en CNF.

El problema a ser resuelto por los alumnos mediante SAT-solvers fue el del conocido Sudoku, que es un juego de ingenio en el cual se da una grilla de 9X9 celdas con algunos números entre el 1 y el 9 ya puestos, y de lo que se trata es de completar la grilla con enteros del 1 al 9 de modo que no se repitan ni en las filas ni en las columnas ni al interior de los nueve sub-cuadrados de 3X3 que hay en la grilla. Los alumnos modelaron este problema, llegando a una forma normal conjuntiva que expresa todas las características del juego de Sudoku.

5. SOLUCION DADA POR LOS ALUMNOS

Para codificar Sudoku como un problema de satisfacibilidad, los alumnos llegaron al siguiente modelo: para cada celda de la grilla S de 9×9 , se asocian 9 posibles valores, así tenemos $9 \times 9 \times 9 = 729$ variables proposicionales. Se asume la notación S_{xyz} para referirse a estas variables, en donde x representa una fila de S , y representa una columna de S y z representa el valor de la celda. Por ejemplo, S_{254} significa que la celda de la fila 2 y columna 5 tiene el valor 4, es decir, $S[2, 5] = 4$.

La forma de plantear la notación y las reglas del Sudoku como CNF es la siguiente:

1. Hay al menos un número en cada celda:

$$\bigwedge_{x=1}^9 \bigwedge_{y=1}^9 \bigvee_{z=1}^9 S_{xyz}$$

2. Cada número aparece a lo más una vez en cada fila:

$$\bigwedge_{y=1}^9 \bigwedge_{z=1}^9 \bigwedge_{x=1}^8 \bigwedge_{i=x+1}^9 (\neg S_{xyz} \vee \neg S_{iys})$$

3. Cada número aparece a lo más una vez en cada columna:

$$\bigwedge_{x=1}^9 \bigwedge_{z=1}^9 \bigwedge_{y=1}^8 \bigwedge_{i=y+1}^9 (\neg S_{xyz} \vee \neg S_{xiz})$$

4. Cada número aparece a lo más una vez en cada sub-grilla de 3×3 :

$$\bigwedge_{z=1}^9 \bigwedge_{i=0}^2 \bigwedge_{j=0}^2 \bigwedge_{x=1}^3 \bigwedge_{y=1}^3 \bigwedge_{k=y+1}^3 (\neg S_{(3i+x)(3j+y)z} \vee \neg S_{(3i+x)(3j+k)z})$$

$$\bigwedge_{z=1}^9 \bigwedge_{i=0}^2 \bigwedge_{j=0}^2 \bigwedge_{x=1}^3 \bigwedge_{y=1}^3 \bigwedge_{k=x+1}^3 \bigwedge_{l=1}^3 (\neg S_{(3i+x)(3j+y)z} \vee \neg S_{(3i+k)(3j+l)z})$$

Esta codificación consta de una fórmula CNF de 8.829 cláusulas, de las cuales 81 son nueve-arias (restricción 1) y 8.748 son binarias (restricción 2 al 4). Como resultado de esta implementación se obtiene un archivo que contiene todas estas restricciones. El formato de ese archivo es el siguiente:

```
p cnf 729 8.829
111 112 113 114 115 116 117 118 119 0
-111 -112 0
-111 -113 0
-111 -114 0
...
```

Esta notación significa que la codificación está en cnf, consta de 729 variables y 8.829 cláusulas. La primera cláusula dice que S_{111} y S_{112} no pueden ser verdaderas al mismo tiempo, es decir que la casilla (1,1) no puede tener al mismo tiempo los valores 1 y 2, etc.

Una vez ingresado a un SAT-solver, en este caso MiniSAT, la respuesta obtenida es:

SAT

-1 -2 ... -99 -100 ... -541 542 -543 ... 999 0

En estos dos archivos, los números que están negados significan que Sxyz es falso. Para el archivo de salida, los valores positivos son los valores que se consideran como respuesta para completar el tablero. Además, la primera línea del archivo de salida con la palabra SAT significa que el problema es satisficible y se entrega la valuación que lo hace verdadero. De lo contrario, se obtiene solamente UNSAT.

6. DISCUSION DEL EJEMPLO

Sin necesidad de dar detalles técnicos extra, o de entrar en temas teóricos relacionados con la teoría de la NP-completitud en ciencia de la computación, lo que queremos destacar aquí es el hecho de que los estudiantes, al desarrollar un ejemplo práctico en el cual efectivamente un problema de carácter computacional es llevado a una forma normal conjuntiva que entrega (cuando es satisficible) una solución al problema original, pueden captar la idea de la reducción de problemas a SAT en forma más “vivencial” por así decirlo que si meramente se expusieran los conceptos en la pizarra o en las transparencias (un poco la idea del proverbio chino que mencionábamos). El alumno efectivamente desarrolla una transformación concreta de un problema a otro, *reflexiona* sobre el tema, es decir que no solamente lo lee, y por lo tanto nos parece que el proceso de aprendizaje resultante es más eficiente y tal vez más humano. Nos comentan los alumnos involucrados que este proceso de reflexión los hace en algunos casos tratar de visualizar otros problemas de otros tipos como un problema a ser planteado mediante una forma normal conjuntiva.

7. CONCLUSION

Dimos en este artículo una reseña del método de aprendizaje basado en resolución de problemas y sugerimos formas de aplicación práctica de este método que nos parecen más adaptadas a la realidad local que la propuesta original del método. Los alumnos responden, se sienten motivados en general, y quedan con un sentimiento positivo de *logro*. El proceso tiende a ser más eficiente en general, y en la terminología de la metáfora inicial, se trata para los docentes de ser más “hormigas” y menos “abejas”.

BIBLIOGRAFIA

- [1] Teaching in tandem-running ants. Franks, N. & T. Richardson. Nature 439, 153. 2006.
- [2] Inductive Teaching and Learning Methods: Definitions, Comparisons and Research Bases. Michael J. Prince & Richard M. Felder. Journal of Engineering Education, 2006.
- [3] Engineering Education - Is Problem-Based or Project-Based Learning the Answer? Julie Mills & David Treagust. Australasian Journal of Engineering Education, 2003-04.
- [4] The Complexity of Theorem-Proving Procedures. Stephen A. Cook, Proceedings of the 3rd IEEE Symposium on the Foundations of Computer Science, pp. 151-158, 1971.