Revista Brasileira de Informática na Educação – RBIE
Brazilian Journal of Computers in Education
(ISSN online: 2317-6121; print: 1414-5685)
https://sol.sbc.org.br/journals/index.php/rbie

Submission: 30/12/2025; 1st round notif.: 28/02/2026; New version: 28/03/2026; 2nd round notif.: 20/04/2026
Camera ready: 04/05/2026; Edition review: 15/05/2026; Available online: 15/05/2026; Published: 15/05/2026

Espacios mentales en el aprendizaje de la programación, un estudio desde la didáctica de la informática

Mental spaces in programming learning, a study from the didactics of computer science

Alejandro Miños Fayad
Departamento de Informática
Instituto Normal de Enseñanza Técnica
Consejo de Formación en Educación
ORCID: 0000-0002-9988-1682
alejandromifa@gmail.com

Juan Valle-Lisboa
Instituto de Biología, Facultad de Ciencias & Centro Interdisciplinario en Cognición para la Enseñanza y el Aprendizaje
Universidad de la República (UDELAR)
ORCID: 0000-0002-5662-5937
juancvl@fcien.edu.uy

Resumen

Para construir un programa informático que resuelva un problema, este debe modelarse técnicamente. Previamente a dicho modelado técnico, se debe elaborar un modelo cognitivo, es decir, identificar y relacionar los principales elementos constitutivos del problema que guíen la construcción de la solución y representen su semántica. Este trabajo de tipo exploratorio, analiza las características del modelado cognitivo realizado por estudiantes de secundaria, el cual se expresa mediante la escritura en lenguaje natural. Para analizar el modelo cognitivo de los estudiantes, se emplearon herramientas estadísticas, así como algoritmos de inteligencia artificial que muestran las relaciones semánticas del discurso. Concluimos que aquellos discursos que hacen referencia a los elementos estructurantes del problema de forma coherente se asocian con programas viables; modelos que también muestran relaciones semánticas complejas. Por otro lado, las construcciones lingüísticas empobrecidas, que no hacen referencia a elementos disciplinarios fuertes y por tanto representan un modelo pobre, generalmente se asocian soluciones deficientes.

Palabras clave: *Didáctica de la informática; Modelos cognitivos; Programación; Computación.*

Abstract

To build a computer program that solves a problem, it must be technically modeled. Prior to such technical modeling, a cognitive model must be developed, that is, the main constituent elements of the problem must be identified and related, guiding the construction of the solution and representing its semantics. This exploratory work analyzes the characteristics of cognitive modeling performed by high school students, which is expressed through writing in natural language. To analyze the students' cognitive model, statistical tools were used, as well as artificial intelligence algorithms that show the semantic relationships of the discourse. We conclude that those discourses that coherently refer to the structuring elements of the problem are associated with viable programs; models that also display complex semantic relationships. On the other hand, impoverished linguistic constructions, which do not refer to strong disciplinary elements and therefore represent a poor model, are generally associated with deficient solutions.

Keywords: *Computer science didactics; Cognitive models; Programming; Computing.*

Cite as: Miños Fayad, A., & Valle-Lisboa, J. (2026). Espacios mentales en el aprendizaje de la programación, un estudio desde la didáctica de la informática. Revista Brasileira de Informática na Educação, vol. 34, pp. 577-597. https://doi.org/10.5753/rbie.2026.7331

1 Introducción

Además de una forma de comunicación, el lenguaje es un instrumento del pensamiento (Chomsky, 1957), por lo que el estudio de cómo los individuos expresan sus ideas y perspectivas puede emplearse para el estudio de los procesos de pensamiento que un individuo sigue al resolver un problema. Cada una de las alocuciones o producciones textuales que realiza un individuo contiene información semántica que representa ideas, hechos o percepciones.

La semántica es el significado lingüístico de un enunciado (Frawley, 2013) y no la mera sumatoria de las semánticas de cada una de las frases o palabras. Además de la composicionalidad, en la que las reglas de combinación de las palabras y su significado generan una interpretación básica, los textos tienen diferentes niveles semánticos, empezando por el nivel de microestructura pero llegando al nivel macroestructural que representa aspectos generales del discurso (Van Dijk, 1980) (Kintsch y van Dijk, 1978). Un texto o un discurso contienen de manera latente la información conceptual que forma parte del conocimiento del hablante. Este conocimiento se organiza en modelos mentales (Johnson-Laird, 2010). Las experiencias personales dan lugar a lo que se denominan modelos mentales, que procuran dar respuestas coherentes a las experiencias del individuo (Van Dijk, 1980). Esos modelos permiten que el individuo haga una interpretación de cada término o palabra, al que le asigna propiedades y características en función de su experiencia representada en el modelo. Una forma de representar y teorizar acerca de esos modelos mentales es mediante espacios mentales. El concepto de espacio mental implica vínculos semánticos entre términos, palabras o frases en los que la distancia se relaciona a la similitud semántica, algo también utilizado en los espacios semánticos que son la base de los modelos computacionales de procesamiento del lenguaje natural. Existen diversos algoritmos para obtener espacios semánticos. El análisis semántico latente permite identificar la relación semántica entre palabras en un contexto (Kintsch, 1998), pudiendo usarse también algoritmos como Word2Vec (Mikolov et al., 2013). Por tanto los espacios semánticos de un individuo no son otra cosa que la representación de las relaciones existentes entre un conjunto de conceptos.

La verbalización o escritura de la solución a un problema implica poder organizar un discurso coherente, que sigue una trayectoria fluida entre los distintos sub temas que conforman la solución (Valle-Lisboa et al., 2014). Dicho de otro modo, el sujeto planifica el discurso, prevé la solución al problema y relaciona las principales ideas que dan lugar a la solución. La escritura es en general más controlada y planificada que la verbalización (Van Dijk, 2019), por lo que el uso de la misma sería un indicador particularmente adecuado para analizar el proceso de planificación del discurso, que a su vez es una interpretación basada en el conocimiento del problema y su descripción (Johnson-Laird, 2010). Todo esto sugiere que el estudio de las producciones lingüísticas de los sujetos podría acercarse a obtener información relevante de los procesos cognitivos que subyacen a la resolución de problemas.

Por tanto el uso de herramientas de análisis semántico permitiría determinar qué tan adecuada es la construcción de una solución, representación o modelo de un problema, debiendo considerarse tres aspectos: la superposición semántica entre aquello a explicar y cómo es explicado; la identificación de los principales conceptos del problema en un discurso coherente; y la relación entre la capacidad de explicar el problema y el dominio disciplinar de parte del individuo.

2 Pensamiento computacional, programación y cognición

Si bien existen variaciones en la definición de qué es el pensamiento computacional existe una suerte de consenso en que el mismo implica hacer uso de la descomposición y abstracción como formas de trabajo (Shute et al., 2017), al tiempo que Wing sostiene que el mismo es el proceso de pensamiento involucrado en plantear y resolver problemas (Wing, 2010). La diferencia entre programación y pensamiento computacional es sutil, pues aunque la segunda no requiere de la primera, la primera es la mejor forma de operativizar la segunda (Webb et al., 2017).

El aprendizaje de la programación permite el desarrollo del pensamiento computacional (Resnick et al., 2009), con efectos cognitivos positivos en la planificación, resolución de problemas o razonamiento lógico entre otros aspectos (Liao y Bright, 1991). Aho sintetiza la relación entre computación y pensamiento computacional de la siguiente forma: "computation is a process that is defined in terms of an underlying model of computation and computational thinking is the thought processes involved in formulating problems so their solutions can be represented as computational steps and algorithms" (Aho, 2012, p. 834).

Programar implica planificar posibles soluciones, siendo por tanto una actividad exigente cognitivamente. La relación entre la programación y las representaciones cognitivas se evidencia principalmente en las habilidades de resolución de problemas (Liao y Bright, 1991) (Brennan y Resnick, 2012) así como en la relación con el lenguaje natural (Maloney et al., 2008; Scaffidi y Chambers, 2012; Fedorenko et al., 2019; Prat et al., 1997). Comprender cómo un individuo construye representaciones cognitivas de un problema podría arrojar luz sobre cómo los mismos resuelven el mismo. Esto último es de importancia educativa, pues estaría también asociado a comprender formas de desarrollar el pensamiento computacional y la enseñanza de la programación.

3 Didáctica de la informática, programación y modelos

La didáctica en el sentido dado por Litwin (1997) está condicionada por el saber disciplinar (Pesce, 2014). En el caso de Informática, su didáctica para enseñanza media tiene como sus elementos estructurantes la computadora, el modelado y la programación (Miños Fayad, 2017); aspectos que determinan las estrategias áulicas, las líneas de investigación y estudio de la disciplina.

Henklain et al. (2025) realizan un meta análisis sobre las dificultades identificadas en la enseñanza de la programación, reconociéndose algunos temas recurrentes: la sintaxis y semántica de los lenguajes, la resolución y formulación de problemas y la identificación de errores, entre otros. El paradigma y lenguaje de programación con el cual se dicta el primer curso de programación es una de las líneas de investigación asociada a su enseñanza, condicionando las formas de resolver o modelar un problema (Clements y Gullo, 1984; Wiedenbeck y Ramalingam, 1999; Fernandez et al., 2002; Grandell et al., 2006; Koulouri et al., 2014). Desde el rol docente es fundamental tener en cuenta las estrategias áulicas, las que se relacionan con la resolución de problemas. En relación a las estrategias áulicas, se consideran entre otras el aprendizaje de conceptos y su combinación (Lahtinenet al., 2005), la incidencia del lenguaje natural como mediador en los procesos de enseñanza y aprendizaje (Sentance y Waite, 2021; López-García et al., 2025), las estrategias inductivas (Robins et al., 2003; Miños Fayad, 2016a), la contextualización de las actividades (Villalobos y Calderón, 2009; Bouvier et al., 2016) o los modelos mentales (Sanders et al., 2006; Mazumder y Pérez, 2024). El estudio la construcción de planes viables es de larga data (Soloway, 1986; Ebrahimi, 1994), algo que también ocurre con la semántica del problema (Winslow, 1996; Xie et al., 2018). Recientemente

se ha incorporado a la investigación el uso de inteligencias artificiales generativas, cómo las mismas podrían influir en los procesos formativos y aspectos éticos asociados a su uso (Piccolo et al., 2023; Prather et al., 2023; Kazemitabaar et al., 2025).

Un aspecto relacionado con la construcción de planes lo conforma el modelado del problema. Un modelo es una representación de aquello a resolver, en el cual se relacionan sus elementos constitutivos y que está condicionado por el tamaño del mismo y quien modela. El modelado es una actividad cognitiva que se encuentra entre la comprensión de la consigna y el modelado técnico disciplinar. Por tanto, crear un modelo es comprender el problema e identificar las relaciones existentes entre sus elementos constitutivos, pudiendo dar lugar luego a un modelo técnico.

En un proyecto de gran porte, el modelo podrá estar en función de la estructura de componentes o paquetes (Pressman, 2010), distanciándose significativamente del algoritmo. Cuando el tamaño del problema es reducido, el modelo es esencialmente el algoritmo a resolver, pudiendo incluso relativizarse si existe como tal. El modelo no es la elección de tipos de datos, objeto, algoritmo, o incluso la descripción precisa de las sentencias a usar. Como para problemas de tamaño reducido el modelo puede coincidir con la descripción del algoritmo, solo es posible identificarlo cuando el mismo se aleja del algoritmo y no es representado como este último. El modelo identifica y relaciona los elementos relevantes del problema, siendo por tanto un espacio semántico (Valle-Lisboa et al., 2014) limitado por los modelos mentales (Van Dijk, 1980).

4 Diseño metodológico

4.1 Problema de investigación

Existen varios artículos que analizan la relación entre la semántica de un problema y su solución (Xie et al., 2019; Fitzgerald et al., 2002; Renumol et al., 2010; Brennan y Resnick, 2012). Sin embargo, no se encontraron trabajos que estudian la construcción de modelos en el sentido dado en este artículo, afirmando Lye y Koh que sería interesante analizar cómo los estudiantes verbalizan el proceso cognitivo (Lye y Koh, 2014). Este trabajo, de tipo exploratorio, considera una grupo de estudiantes y analiza cómo los mismos construyen modelos para resolver los problemas planteados; aspecto que tiene relación con el aprendizaje de la programación, el pensamiento computacional y los procesos cognitivos intervinientes.

De este modo queda definido el problema de investigación de la siguiente manera: ”Para construir un programa primero se debe modelar el problema a resolver, esto es, construir una representación cognitiva de alto nivel que relaciona convenientemente las principales partes del problema y es expresada mediante lenguaje natural. Con el fin de maximizar los aprendizajes en programación, resulta relevante comprender cómo los estudiantes modelan un programa.”

Puesto que no se identificaron trabajos que estudia cómo es el modelado de programas, en el sentido dado en este trabajo, es que el mismo fue de tipo exploratorio. Aunque este trabajo no establece hipótesis, si se presentan preguntas de investigación las que quedan planteadas de la siguiente forma:

1. ¿Cómo expresan los estudiantes el modelo de un problema a resolver?
2. ¿Qué relación hay entre los elementos constitutivos del problema y la forma de expresarlo por parte de los alumnos?
3. ¿Qué relación existe entre las soluciones construidas por los estudiantes y la forma como expresado en lenguaje natural el modelado del problema?

4.2 Población y actividades

La población estudiada fueron estudiantes de 2º año de enseñanza media básica, 59 varones y 35 mujeres, entre 13 y 15 años, en Montevideo – Uruguay.

La primera actividad consistió en la formulación de preguntas, las respuestas se asociaron a un conjunto de variables:

"Cuando vas a hacer un programa, ¿cómo te das cuenta que debes usar una variable?" (R_VARIABLES).

"¿Cómo te das cuenta que debes usar una estructura de repetición?" (R_REPETICIÓN).

"¿Cómo te das cuenta que debes usar una instrucción Si?" (R_CONDICIÓN).

Posteriormente se dio la imagen Figura 1 y la consigna:

"Se quiere programar un juego en el cual se controla el movimiento de un auto para que se mueva por un circuito. El salir del mismo el auto se destruye y finaliza el juego. El auto avanza solo de a 2 pasos. Los puntos se logran al pisar con el auto una serie de marcas en el circuito, obteniéndose 10 puntos por cada marca. Al llegar al fin (marcado con una línea) se indica que ganó y finaliza el juego mostrando los puntos".

Se solicitó a los estudiantes: "cuéntale a alguien cómo resolverías el programa siguiente", identificándose 4 variables: T_OBJETOS (relación entre objetos y existencia de un vehículo), T_MOVIMIENTOS (explicación sobre el movimiento de objetos), T_VARIABLES (identificación y operación de la misma) y T_CONDICIONES (existencia y eventos con condiciones).

Las respuestas fueron evaluadas y asignaron 0, 1 o 2 como valor en función de la correctitud de la respuesta: 0 para respuestas incorrectas o con baja coherencia; 1 respuestas parcialmente correctas o que ejemplificaban el uso sin hacer referencia a los elementos constitutivos de las misma; 2 se asignó a las respuestas correctas o con errores no relevantes. Por ejemplo, R_VARIABLES tomó el valor 2 si el estudiante respondía que había que almacenar un valor para ser usado posteriormente; si el alumno respondía "cuando tenés que sumar puntos" o "cuando en un juego tenes que guardar los puntos" se asignaba 1; 0 si se respondía "para jugar" o "para cambiar datos".

Un único evaluador fue el encargado de la asignación de puntos para cada una de las preguntas. Dada la cantidad de preguntas y respuestas, se tomó una pregunta y evaluaron todas las respuestas asociadas. Con el fin de procurar evitar que unas respuestas afecten la evaluación de otras asociadas a otra pregunta, es que se evaluaron las respuestas a una pregunta sin hacer lo mismo con otras.

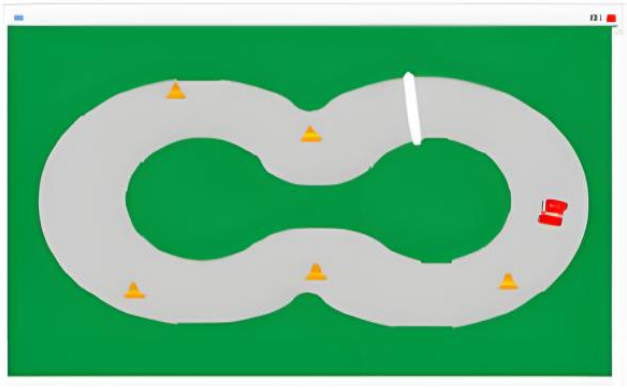

Figura 1: imagen entregada a los estudiantes.

Dadas las características de Scratch (Resnick et al., 2009) (Maloney et al., 2008) (Maloney et al., 2010), es que se usó este lenguaje para la construcción del programa.

La tercera actividad consistió en la programación individual en Scratch del problema, calificada de 1 a 12 (NOTA). Se agruparon los estudiantes según la calificación: ALTA_N (12 a 9), MEDIA_N (8 a 5) y BAJA_N (4 o menos). Las variables SUMA_R y SUMA_T fueron la suma de las variables R y T respectivamente. SUMAR_T sería un indicador de la coherencia y correctitud de las respuestas como un todo, asociado esto a la semántica del problema para el problema y SUMA_R para los conceptos teóricos.

4.3 Métodos y técnicas de análisis

Dado que este trabajo fue de tipo experimental, es que se utilizaron múltiples técnicas de procesamiento de datos con el fin de tener una visión global del modelado que hacen los estudiantes.

Para cada estudiante se realizó la corrección de los programas y los enunciados sobre cómo resolver el problema y las preguntas teóricas.

También se determinó la cantidad de palabras de los enunciados y las relaciones entre ellos. Se usaron funciones estadísticas con el fin de identificar las eventuales relaciones de este tipo, aplicándose test de confiabilidad a los resultados.

El software Speech Graphs (Mota, Furtado, Maia, Copelli y Ribeiro, 2014) toma un enunciado y lo representa como un grafo, donde cada nodo es una palabra y la relación entre dos se representa con una arista. Se usó este software para procurar identificar regularidades de cada uno de los enunciados, así como para obtener estadísticas asociada a la cantidad de palabras y la relación entre ellas (variables NODOS y ARISTAS). Se calculó el porcentaje de exceso de aristas (PEA) como ARISTAS / NODOS – 1; variable que podría estar asociada con la complejidad de las relaciones semánticas.

El algoritmo K – Means toma un conjunto de registros, conformado por varios campos y los agrupa en función de la distancia que tienen al centroide; el cual a su vez está asociado a la cantidad de clusters. De este modo K – Means agrupa los registros que mantienen alguna similitud en función de las variables constitutivas. El algoritmo asegura que el agrupamiento no es forzado por métricas o la intervención del investigador.

Se derivaron representaciones vectoriales a partir del algoritmo de aprendizaje no supervisado Word3Vec, el cual cuantifica las relaciones semánticas en un texto. Dado un texto, diccionario o corpus textual, el algoritmo cuantifica la relación semántica entre palabras del mismo, de modo que cuanto mayor es la relación semántica entre las mismas mayor es el valor numérico asociado. Se construyó un diccionario para cada grupo de calificaciones, conformado por la explicación sobre cómo resolver el problema y las respuestas teóricas.

5 Resultados obtenidos

5.1 Descripción general

Evaluados los trabajos la cantidad de estudiantes que obtuvieron ALTA_N, MEDIA_N y BAJA_N fue similar, 30, 31 y 33 estudiantes respectivamente.

Considerando los enunciados construidos para explicar el problema, en promedio cada uno de ellos requirió 39 palabras distintas y 57 relaciones entre ellas. La relación entre nodos (palabras) y aristas (relaciones entre palabras) del grafo representación del enunciado es directa, con un R^2 de 0,953, en decir, más palabras se relaciona con una mayor relación entre ellas. La

cantidad de palabras promedio no está asociada a la nota de 1 a 12. Sin embargo, cuando se considera la cantidad total de palabras de cada enunciado por grupo de notas, se observa que ALTA_N tiene más palabras que MEDIA_N y esta que BAJA_N. Esto mismo aplica para los enunciado lematizados

La PEA promedio resultante fue de 35%, donde los estudiantes con calificación ALTA_N obtuvieron PEA promedio de 44,5%, para MEDIA_N 35,4% y con calificaciones BAJA_N la PEA promedio fue 27,1%.

Los enunciados de los estudiantes incluyeron múltiples faltas de ortografía al tiempo que se usaron varias palabras para hacer referencia al mismo concepto o procedimiento (auto, coche, vehículo; cono, conito, conos, etc.). Con el fin de evitar que esto afecte el análisis semántico, se procedió a normalizar el texto: eliminando tildes, corrigiendo las faltas de ortografía y usando un mismo término para definir otros tantos.

5.2 Relaciones entre el grafo discurso y las variables relevadas

Existe una relación directa muy fuerte entre la cantidad de nodos y aristas en los grafos del discurso de los estudiantes. El R2 resultante fue del 95 %, en tanto que las pruebas t y Fischer dieron correlaciones mayores al 0,97 con $p<0,01$.

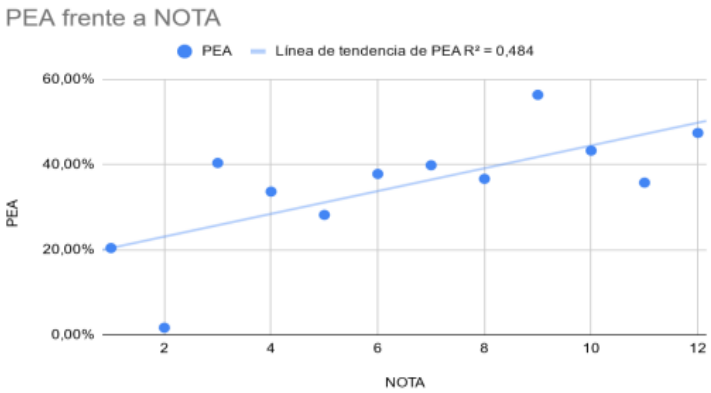

Figura 2: Relación entre PEA y NOTA.

Figura 2 muestra la relación fuerte entre los resultados obtenidos en la programación (NOTA) y PEA, lo que estaría indicando una relación directa entre la producción escrita y la calificación obtenida considerando el PEA promedio por calificación. Sin embargo, cuando consideramos el R2 entre cada PEA y NOTA, el mismo es de 0,07.

La correlación Pearsons entre NOTA y las variables, SUMA_R y PEA, para las categorías ALTA_N, MEDIA_N y BAJA_N, ha evidenciado ser no significativos al 95% para los test t y Fisher; la correlación entre NOTA y SUMA_T es del 0,555 al 99%. La ausencia de una alta correlación puede ser producto de la baja linealidad y acentuarse en poblaciones relativamente reducidas como son los tres grupos de notas.

El R2 entre SUMA_T y las variables NOTA, NODOS, ARISTAS y PEA dieron 0,308, 0,371, 0,345 y 0,264 respectivamente (significativas al 99%). Las correlaciones entre SUMA_T por un lado y NOTA, NODOS, ARISTAS y PEA fueron de 0,555, 0,609, 0,587 y 0,513 respectivamente, significativas al 95%. El ajuste de las pruebas por Bonferroni, entre SUMA_T y cada una de las variables, dio un valor de p de 4,9%, manteniendo las correlaciones su significancia.

El test Chi cuadrado de Pearson con los datos de Tabla 1 dio como resultado 35,305, p=4,021 x 10-07 y df 4; por lo que SUMA_T está relacionada con los resultados obtenidos en la programación, asociado a las categorías ALTA_N, MEDIA_N y BAJA_N y la relación no es aleatoria.

Tabla 2 muestra las correlaciones entre SUMA_T y NODOS, ARISTAS y PEA (*p al 95%; **p al 99%.), agrupadas según las calificaciones; observando una relación directa entre la correlación y las calificaciones obtenidas. Aplicando Bonferroni, el nuevo valor de p pasa a 4,39%, manteniendo todas las correlaciones su significancia.

Tabla 1: Cantidad de estudiantes según SUMA_T y resultados.

SUMA_T	ALTA_N	MEDIA_N	BAJA_N
8 a 6	19	7	2
5 a 3	4	13	7
2 a 0	6	11	25

Tabla 2: Correlación entre SUMA_T y NODOS, ARISTAS y PEA.

	NODOS	ARISTAS	PEA
ALTA_N	0,641**	0,624**	0,623**
MEDIA_N	0,595**	0,573**	0,499**
BAJA_N	0,529**	0,487**	0,358**

Tabla 3: Cantidad de estudiantes según SUMA_R y resultados.

SUMA_R	ALTA_N	MEDIA_N	BAJA_N
6 a 4	9	4	4
3 a 2	13	13	10
1 a 0	8	14	19

La cantidad de estudiantes para cada grupo de calificaciones según los rangos de SUMA_R se encuentran en Tabla 3. En general los bajos resultados de SUMA_R se asocian con estudiantes de grupos que no son ALTA_N. El test Chi cuadrado Pearsons dio como resultado 7,063, con p 0,1326 y df 4, por lo que las variables analizadas no están relacionadas y son independientes.

5.3 Enunciados, lematización y conceptualización

Todo proceso de evaluación requiere identificar criterios de correctitud, no siendo este trabajo una excepción. Por tanto, se pidió a un grupo de profesores de informática que expresen cómo sería para ellos una redacción correcta del problema. Todos los profesores explicaron que un vehículo debería moverse por un circuito y que ocurrían una serie de acciones cuando este tocaba cono. Dos aspectos importantes surgieron de las respuestas: el uso de distintos términos para hacer referencia al mismo objeto (por ejemplo: auto, vehículo, coche, etc.); y la existencia de ideas comunes en la descripción (por ejemplo: movimiento de un vehículo, acciones al tocar los conos o la línea de meta). Esto dio lugar a agrupar las palabras con similitud semántica, en el contexto de la respuesta, así como identificar los elementos que debe tener una respuesta correcta. Este grupo de palabras son los denominados conceptos. Por tanto queda generada una respuesta general, la cual se redacta en función de los conceptos fuertes (identificados por mayúsculas y donde un mismo grupo de palabras se asocia al mismo concepto fuerte):

El Auto / Coche se programa para que al Tocar un Cono / Conos / Conito, que es Anaranjado / Naranja, o un Objeto / Personaje del juego, se Modifique / Cambie el valor de la

Variable al Sumar uno al valor total. La Variable Cuenta / Contaría / Contabiliza los Puntos / Puntaje. Cuando el Auto / Coche sale del circuito es porque Toca / Tocaría / Tocase el Color Verde, Cambiando / Modificando el Disfraz, lo cual se puede hacer Desapareciendo / Ocultando / Escondiendo el Objeto / Personaje y Finalizando / Terminando el juego.

Se puede observar que un mismo concepto, como por ejemplo el vehículo que se mueve por el escenario, se asocia con Vehículo, Auto o Coche. Como dijimos previamente, el concepto identificado se puede asociar a varias palabras, las cuales permitirían identificar el objeto y relacionarlo con otros aspectos del juego. La propia definición de conceptos tiene un grado de subjetividad que solamente puede ser dirimida en función de la realidad a resolver, las características de los alumnos, su lenguaje y la percepción del evaluador. Es por esto que la asociación entre palabras y conceptos fuertes no es prescriptiva a priori, pudiendo incluirse en un grupo de palabras asociadas a un concepto fuerte otras que cumplen la misma función en el discurso de los estudiantes. Los conceptos agrupados fueron: Auto / Coche; Cono; Color; Objeto / Personaje; Condición; Variable; Naranja; Verde; Blanco; Línea / Meta; Sumar; Punto; Combinar / Modificar; Tocar; Final / Terminar; Contar / Conteo / Cuenta; Disfraz / Desaparecer / Ocultar / Esconder

Se analizó el grafo no dirigido resultante de los enunciados propios del modelado del problema: discurso completo, lematizado y de conceptos fuertes. La Figura 3 muestran dos grafos de ejemplo de enunciados completos; en ninguno de los tres tipos de enunciados se identificó alguna regularidad. Si bien en ambos grafos se podría afirmar que muestran discursos complejos, el PEA muestra valores distintos.

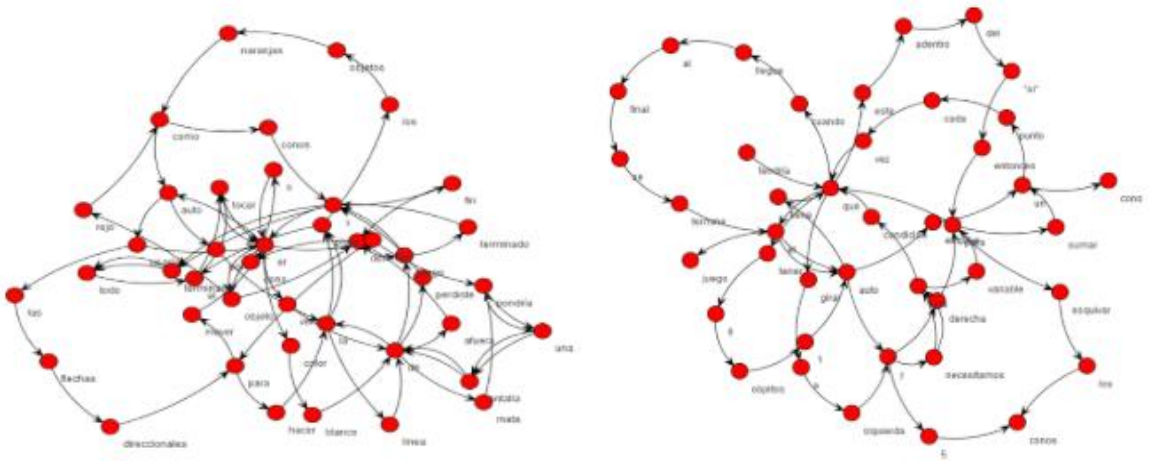

Figura 3: discurso de dos estudiantes con notas 12 y 10; PEA 71,43% y 28,57 y 42 nodos.

Para todas las categorías de palabras, en Tabla 4 se evidencia que los estudiantes que logran mejores calificaciones usan más palabras que aquellos que tienen peores calificaciones, lo que podría estar sugiriendo una mayor riqueza en el uso de lenguaje. Otra interpretación es que los estudiantes de ALTA_N requieren la construcción de enunciados complejos para representar el problema, mientras que el grupo BAJA_N usa menos palabras o tiene mayor dificultad para modelar el problema.

Tabla 4: Cantidad de palabras según nota y enunciado

SUMA_T	ALTA_N	MEDIA_N	BAJA_N
COMPLETO	2120	1760	1704
LEMATIZADO	1156	938	939
CONCEPTOS	475	389	294

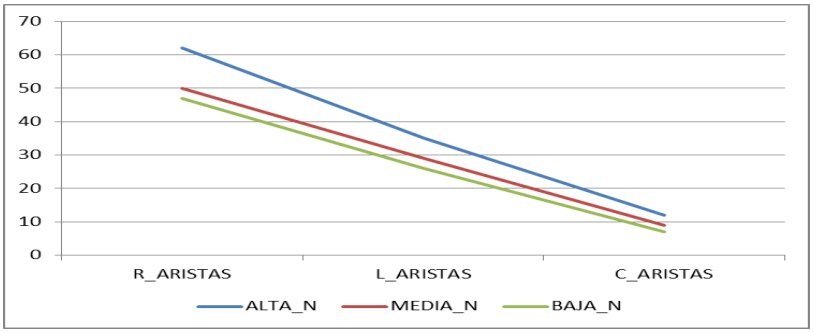

Figura 4: Aristas por tipo de enunciado y grupo de calificación.

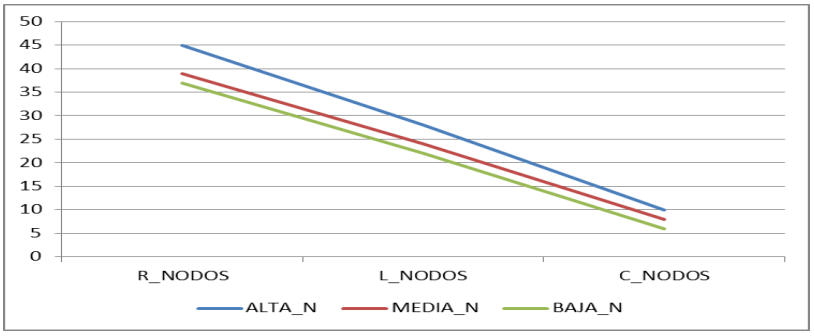

Figura 5: Nodos por tipo de enunciado y grupo de calificación.

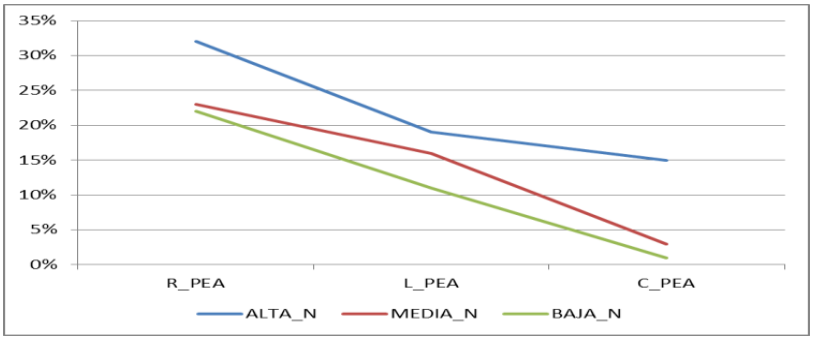

Figura 6: PEA por tipo de enunciado y grupo de calificación.

Las figuras 4, 5 y 6 muestran la relación directa entre los resultados de cada variable y el grupo de calificaciones.

5.4 Clusterización de datos

Ejecutado el algoritmo de clusterización se relacionó el resultado obtenido con las calificaciones de los alumnos. En todos los casos se usó como conjunto de datos la población completa, independientemente de la calificación.

La elección de la cantidad de clusters se basó en la dispersión de los datos generados y en las regularidades observadas. Se observó que con 5 clusters el solapamiento entre los mismos se reducida en relación a trabajar con menos, al tiempo que fue posible identificar regularidades.

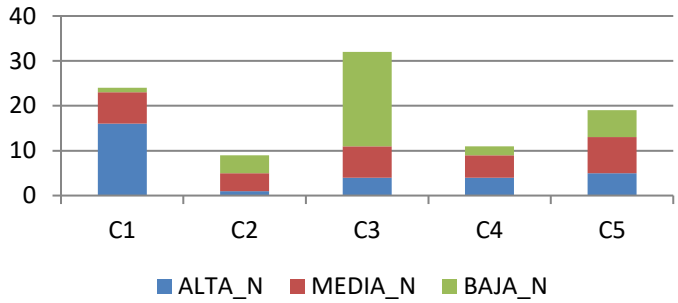

Figura 7: Cantidad de estudiantes y grupo de calificaciones por cluster y considerando todas las variables T_MOVIMIENTOS, T_OBJETOS, T_VARIABLES y T_CONDICIONES.

La Figura 7 muestra los resultados obtenidos del proceso de clusterización con las variables. T_OBJETOS, T_MOVIMIENTOS, T_VARIABLES y T_CONDICIONES. Los estudiantes con mejores y peores calificaciones se concentran en el cluster 1 y 3 respectivamente. El promedio de los valores SUMA_T por cluster, fue 7, 3, 0, 5 y 3. Por tanto, el primer cluster concentra los mejores resultados de la evaluación de los programas, en tanto que el tercer cluster tiene un promedio de SUMA_T de 0 y concentra los peores resultados.

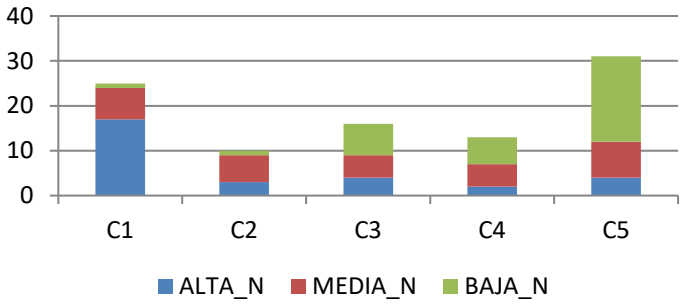

Figura 8: Cantidad de estudiantes y grupo de calificaciones por cluster y considerando las variables T_OBJETOS, T_VARIABLES y T_CONDICIONES

La Figura 8 presenta la composición de los clusters sin considerar la variable T_MOVIMIENTOS. El resultado obtenido es similar al visto en figura 6 en el sentido de agrupar los mejores y peores resultados. Los valores promedio de SUMA_T fueron 6, 3, 2, 3 y 0. El cluster 1 agrupa los estudiantes que tuvieron los mejores resultados (pasando de 16 a 17) y el 5 los peores (disminuyendo de 21 a 19). La escasa incidencia de la variable T_MOVIMIENTOS podría estar indicando una baja incidencia de la misma, no siendo un es parte significativa del modelo.

La clusterización usando todas las variables R_VARIABLE, R_CONDICIÓN y R_REPETICIÓN, o sub conjuntos de ellas, así como sub conjuntos de las variables usadas en la tabla anterior no mostró regularidades. En efecto, no se ha podido identificar clusters que agruparon una cantidad significativa de estudiantes con buenos o malos resultados.

5.5 Análisis de relaciones semánticas

Se consideró que el discurso genérico presentado en 5.3. se ajustaba a una respuesta correcta, por lo que se tomó como referencia para cuantificar la relación entre las palabras fuerte de ese discurso y las relaciones con otras de los grupos de estudiantes.

Los parámetros usados en el algoritmo Word2Vec, además del corpus textual lematizado, y 4 hilos (parámetro workers), fueron:

vector_size = 150. Determina la cantidad de dígitos que permiten representar cada palabra.

window=10. Determina la ventana de contexto. Es decir, dada una palabra la cantidad de tokens (palabras en este caso) que son usadas para capturar relaciones entre las mismas. De este modo, el modelo puede identificar relaciones semánticas de largo alcance

min_count=1. Determina la frecuencia de una palabra para ser incluida en el modelo. Dado que se normalizó semánticamente el corpus textual (mediante corrección sintáctica y sustitución palabras por sus sinónimos), se trabajó con frecuencia 1.

La Tabla 5 muestra las palabras más relacionadas con la seleccionada para los grupos de estudiantes que tienen un modelado alto y bajo; en el caso de los estudiantes del grupo Medio los resultados no son tan evidentes como en estos grupos. Para cada palabra considerada, en el grupo ALTA_N se muestran las palabras más cercanas a la dada y que tienen una relación semántica más fuerte con las primeras. En el caso de los estudiantes del grupo BAJA_N, se observa que la relación es más débil que con ALTA_N.

Tabla 5: Palabras cercanas semánticamente a una dada agrupadas por grupo de calificaciones

	Modelado Alto		Modelado Bajo	
Fin	Terminado	0,839	Toca	0,761
	terminaría	0,822	Partida	0,682
	Donde	0,785	Perdería	0,681
Tocar	Nuevamente	0,807	Opciones	0,775
	Llegar	0,674	Al	0,759
	Salir	0,662	Aparezca	0,744
Color	Finalmente	0,721	Va	0,745
	Blanco	0,691	Objeto	0,701
	aparezca	0,682	Bandera	0,684
Afuera	ejecuta	0,800	Corriendo	0,825
	Circuito	0.722	Seguir	0,790
	Dentro	0,659	Pueda	0,751
Sale	Choca	0,775	Del	0,835
	Descuenta	0,741	Rebote	0,789
	Pista	0,728	Mueve	0,721
Cono	presionar	0,697	Cada	0,828
	Recorrido	0,672	Mostrar	0,751
	Siguiente	0,655	Sumes	0,751
Variable	Sumara	0,720	Creas	0,717
	deberíamos	0,731	Valor	0,694
	Cuente	0.707	Utilizaría	0,663
Punto	Suma	0,657	Empiece	0,773
	Necesitamos	0,634	Línea	0,760
	Cono	0,632	Situación	0,745
Suma	Esquive	0,740	Arriba	0,804
	Punto	0,657	Conos	0,697
	cada	0,656	barra	0,671

6 Discusión

Los alumnos que han obtenido mejores calificaciones son también quienes evidencian un discurso más complejo. Estos enunciados tienden a estar constituidos por más palabras, conceptos fuertes y con mayores referencias a palabras previamente mencionadas (PEA).

El ANOVA realizado sobre los grupos de calificaciones muestra un p-value apenas superior a 0,05. Si bien el resultado no modifica sustancialmente la probabilidad obtenida, podría asociarse a la falta de potencia estadística del experimento.

Otro elemento a ser tenido en cuenta es la coherencia en la explicación de cada una de las ideas fuertes que estructuran la solución del problema. Para cada variable que permite cuantificar SUMA_T se observó que las mismas en general toman valor 2 (y por tanto SUMA_T se acerca a 8) para aquellos alumnos que han obtenido calificaciones altas. Este valor se asigna a enunciados que representan de forma adecuada el concepto a evaluar. Si bien en este trabajo no se usó software que cuantifique y objetivase la cohesión del texto, se observa que las mismas son coherentes, pertinentes y adecuadas. Los estudiantes que tienen los mayores valores de SUMA_T son también quienes expresan mejor el modelo, existiendo una superposición de ideas entre la representación del mismo y la realidad dada (Crossley et al., 2019). La evaluación de las respuestas de los estudiantes, al igual que las propias categorías a corregir, es una debilidad del trabajo, pues la gran cantidad de enunciados a evaluar y su interpretación podrían dar lugar a criterios distintos por parte de cada lector.

Al considerar el discurso de los estudiantes como un todo, usando SUMA_T, se observa que los valores más altos de la variable tienen asociados un mayor número de estudiantes con calificaciones entre 9 y 12, y un también mayor porcentaje de relación entre aristas y nodos. Inversamente, estudiantes con menores valores de SUMA_T presentan también una menor relación entre nodos y aristas, agrupando a una mayor cantidad de alumnos con bajos rendimientos en la actividad práctica. El discurso de los estudiantes (SUMA_T) se relaciona con los resultados obtenidos medidos a través de las calificaciones, no siendo esta relación al azar. Existe correlación estadísticamente significativa y superior al 50% en todos los casos entre SUMA_T y NODOS, ARISTAS y PEA. Esto último se mantiene cuando las correlaciones se calculan en función de los grupos de calificaciones obtenidas. Al observar el comportamiento de los datos en función de las calificaciones, se observa que a mayores notas, también mayor es la correlación. Esto último estaría indicando que quienes tienen mejores notas también tienen mejores discursos desde lo disciplinar, más complejos y elaborados.

Si bien la relación entre NOTA y PEA es baja, no sucede lo mismo entre SUMA_T por un lado y NODOS, ARISTAS y PEA por otro, siendo la correlación superior al 50%; lo que podría estar evidenciando una relación fuerte entre el contenido del enunciado y cómo se construye.

Dado el grafo que representa el discurso de los estudiantes, la relación entre la cantidad de nodos y aristas muestra una asociación positiva con el grupo de calificaciones, evidenciando un discurso con más palabras y relaciones entre ellas. Lo anterior unido a valores más altos de SUMA_T para los estudiantes de calificaciones más altas, estaría en línea con una mayor sofisticación del discurso (Skalicky et al., 2017). Se ha identificado un conjunto de palabras asociadas a conceptos fuertes que hacen al modelado de la solución; encontrándose más frecuentemente entre quienes han obtenido calificaciones altas que en el resto de los estudiantes.

Los valores de SUMA_T sugieren que en general existe una coherencia y cohesión semántica en el grupo de estudiantes de ALTA_N superior al grupo BAJA_N, donde el primer grupo parece haber captado el dominio a modelar (McNamara, 2004), siendo consistente con lo expresado por Allen et al. (2015). La alta y baja coherencia semántica se da en los grupos ALTA_N y BAJA_N respectivamente, quienes a su vez concentran los mejores y peores

resultados de SUMA_T respectivamente, agrupamiento de datos que no es al azar. Todo esto estaría permitiendo afirmar que en general se han construido espacios semánticos más coherentes (Valle-Lisboa y Mizraji, 2007) en los grupos de estudiantes que tienen mejores calificaciones que entre quienes tienen calificaciones más bajas. Si bien los estudiantes con mejores resultados en SUMA_T son quienes tienen mejores respuestas en las preguntas, cabría preguntarse si esto se ve reflejado también en la relación entre los conceptos fuertes del discurso y qué tan fluido es su trayecto en el grafo de ideas (Valle-Lisboa et al., 2014).

Las respuestas a conceptos teóricos básicos agrupadas en SUMA_R, muestra que más del 50% de los estudiantes que tienen malos resultados en la construcción del programa (variable BAJA_N) tienen también pobres respuestas a los conceptos teóricos. El test Chi cuadrado para estos datos muestra que no existe relación entre los resultados obtenidos en las respuestas teóricas y la calificación final. En vista de estos resultados podríamos estar diciendo que las evaluaciones teóricas, que consisten en explicar conceptos básicos de programación, no serían un buen indicador de cómo los estudiantes pueden resolver la construcción de programas viables, aunque sí podrían serlo de la construcción de programas no viables. Cabe preguntarse si las propias preguntas planteadas son adecuadas como predictores de los conocimientos de los estudiantes y si las respuestas son de tipo memorísticas o fruto de un análisis y relación de ideas previas. En efecto, podríamos estar frente a un grupo de preguntas que si bien son relevantes desde la programación y el problema planteado, resultan insuficientes como forma de conocer qué saben los estudiantes. Al mismo tiempo debe tenerse en cuenta que los estudiantes pueden entender que aprender implica reproducir literalmente enunciados; aspectos que podrían dar lugar a respuestas que responden parcialmente a la consigna.

El análisis del discurso completo, lematizado o de conceptos fuertes no modifica la tendencia de los resultados, pues a mayor calificación (o grupo de ella) mayores valores de NODOS, ARISTAS y PEA tienen los alumnos. Una mayor cantidad de palabras en general y de términos fuertes en particular, propio de quienes tienen mejores resultados, estaría mostrando que las mismas se relacionan más fuertemente con las respuestas esperadas, que hacen a un correcto modelado del problema y que deberían estar en consonancia con las respuestas dadas por los profesores. Los estudiantes que han obtenido peores calificaciones muestran una reducción de conceptos fuertes en relación al resto, lo que estaría dando a entender mayores dificultades para representar el problema. Por tanto, se estaría verificando en los estudiantes que obtienen buenos resultados una superposición semántica con la solución esperada, cabiendo preguntarse si los resultados son análogos a los presentados por Crossley et al. (2019).

Realizada la clusterización de datos con todas las variables T, dos agrupamientos se diferenciaron del resto: el que se asocia a las calificaciones y valores de SUMA_T (variable no usada para la clusterización pero sí para analizar el propio cluster) más altas, por un lado; y aquellas que agrupan a las calificaciones y valores de SUMA_T más bajo por otro. Todos los alumnos que han obtenido 0 o 1 en SUMA_T, lo que significa tres o más respuestas evaluadas como 0, se encuentran en el mismo cluster. Casi dos terceras partes del total de alumnos que tienen notas bajas pertenecen al mismo cluster; la misma relación se observa entre quienes tienen calificaciones altas. Considerada la representación que hacen los estudiantes del problema, las mismas tienen relación con los productos efectivamente construidos; por lo que se podría afirmar que sería de esperar que un estudiante que haya presentado un modelo adecuado tiene asociada una calificación alta, e inversamente; y si el modelo no lo es, la calificación sería baja. El cluster en el cual se encuentran estudiantes con calificaciones altas tiene también un valor promedio de SUMA_T alto; inversamente, los cluster que concentran a quienes obtienen peores calificaciones tienen también menores valores promedio de SUMA_T. No considerar la variable T_MOVIMIENTOS en el proceso no afecta significativamente los resultados.

El análisis de las palabras fuertes del discurso evidencia que las mismas forman vectores coherentes, relacionadas semánticamente; desde la programación de la solución, al menos para los estudiantes que han obtenido mejores calificaciones. Dada una palabra que forma parte de la respuesta esperada (como Punto), se observa que quienes obtienen mejores calificaciones presentan otras palabras que también se relacionan fuertemente con otras de la respuesta sobre cómo resolver el problema; esto en general no se da entre quienes han obtenido calificaciones más bajas. Este último grupo de estudiantes muestra resultados menos coherentes, donde las palabras resultado no parecen tener relación significativa con el dominio del problema como era de esperar (McNamara, 2004), ni con el resto de las palabras que conforman el vector de relaciones semánticas, dando la impresión que no han podido construir espacios semánticos coherentes (Valle-Lisboa y Mizraji, 2007) como sí lo han logrado quienes han obtenido calificaciones más altas.

El programa solicitado así como el lenguaje pudo influir en los resultados obtenidos. La consigna en sí misma tenía un grado de concretitud significativo, por lo que se debería considerar si esto no incidió en la sub valoración del modelo por parte de los estudiantes. En tal sentido, algunos adolescentes pudieron entender que la mejor forma de explicar cómo resolver el problema en cuestión es la propia solución, aspecto que fue evaluado como negativo en la tabulación de datos. Otros alumnos pudieron considerar que parte del problema a explicar resultaba sencillo y por tanto que no era necesaria una explicación que fuese más allá una breve descripción. Si bien el extremo anterior es válido para todos los estudiantes, se entiende que es más probable que se haya aplicado a quienes tienen un mejor desempeño en programación, pues aumentaría la probabilidad que consideren triviales algunos contenidos. Es por esto que se podrían estar definiendo tres posibles líneas de trabajo complementarias con el fin de continuar analizando el modelado: pedirle a los alumnos que expliquen todo aquello a resolver; solicitarles explicar cómo resolverían el problema a la vez que se les niega la posibilidad de usar una serie de palabras (Miños Fayad, 2016b); o bien construir una consigna más genérica y menos atada a la solución visual de los estudiantes.

Pedir que los alumnos expliquen todo aquello que se debía hacer permitiría tener un diccionario de datos mayor, con más posibilidades de realizar un análisis semántico de cohesión y coherencia del texto. Sin embargo esto podría implicar también que el alumno, en su afán de complacer al profesor, agregue información no necesaria o poco relevante. Impedir que los alumnos usen una serie de palabras evitaría que estos usen los mismos términos que deben definir (como usar la palabra “condiciones” para definir qué es una condición), obligándolos a un esfuerzo cognitivo que permite reestructurar la respuesta. Como contrapartida debería tenerse en cuenta las posibles dificultades que puedan expresar los alumnos en el uso del lenguaje, los sinónimos y el uso de términos que representen la misma idea de una forma distinta a la pensada originalmente. El trabajo con consignas más amplias evitaría el alto grado de concretización de los programas, construyendo soluciones más genéricas, pudiendo observarse así más claramente si existe o no una trayectoria fluida entre los grandes conceptos del texto (Valle-Lisboa et al., 2014). El esfuerzo de parte del estudiante para representar la solución podría convertirse en un obstáculo, a la vez que permitiría manejar más conceptos y la relación entre ellos.

7 Conclusiones

La explicación sobre cómo resolver un problema computacional es el modelo de dicho problema y se expresó mediante la variable SUMA_T. Este modelo se evidencia mediante el lenguaje natural y hace referencia a los elementos constitutivos del problema y su relación.

Los estudiantes que modelaron mejor sobre cómo sería la solución a realizar obtuvieron las mejores calificaciones en la programación del problema, esta representación es también coherente y adecuada. Los estudiantes que no lograron modelar de forma correcta la solución, son también quienes en general no han podido construir programas viables. Se verifica una relación directa entre los grupos de calificaciones y la cantidad de palabras, las relaciones que se establecen entre dichas palabras (aristas del grafo) y PEA. Los estudiantes con mejores calificaciones presentan relaciones semánticas más complejas, por la cantidad de palabras totales, de conceptos fuertes o las relaciones entre las palabras.

Mayores valores de SUMA_T se asocian a la construcción de mejores programas y calificaciones, pudiendo afirmar que los estudiantes con mejores resultados presentan relaciones léxicas más complejas (Skalicky et al., 2017). Si bien no se estableció un criterio de cohesión del texto en el sentido dado por Allen et al. (2015), es cierto que la evaluación de las respuestas implica considerar aspectos asociados a ella, estando por tanto en condiciones de afirmar que mayores valores de SUMA_T podrían ser también un indicador de la coherencia de la respuesta. De este modo se afirma que existen espacios semánticos claramente definidos en función de las calificaciones obtenidas en el sentido dado por (Valle-Lisboa y Mizraji, 2007). Cabe preguntarse si la relación que se evidencia entre la construcción de buenos resúmenes y el conocimiento del texto original descrito por Crossley et al. (Crossley et al., 2019) se estaría aplicando también en este caso, de modo que quienes representan mejor cómo resolver el problema son también quienes lo han comprendido mejor; pregunta que entendemos debería ser respondida afirmativamente.

Los alumnos con registros más altos de SUMA_T son quienes están logrando identificar los grandes elementos constitutivos del problema en el sentido dado por Soloway (Soloway, 1986). Dada la relación entre SUMA_T y las calificaciones, se estaría verificando lo expresado por Winslow (1996) en el sentido que quienes logran comprender y modelar la semántica del problema a resolver, son quienes también logran hacer uso de la sintaxis adecuada a la solución.

El grupo de estudiantes que obtuvo una puntuación baja en SUMA_T, presentó como errores recurrentes la ausencia de respuestas, su ambigüedad o el uso de ejemplos sin referencia clara al concepto a trabajar. Los resultados obtenidos por los estudiantes de este grupo estarían en consonancia con lo planteado por Meerbaum-Salant et al. (2013), existiendo una tendencia a la simplificación y concretización de los conceptos a usar.

La correctitud de las respuestas sobre conceptos teóricos (SUMA_R) no parece estar relacionada con buenos desempeños en el modelado y la construcción del programa en general. Sin embargo, SUMA_R sí sería un indicador de pobres resultados en la construcción del programa.

Una limitante de esta investigación lo constituye el hecho de trabajar con un solo evaluador. Si bien se establecieron criterios de corrección, esto podría estar condicionándola, por considerar como válidas soluciones o respuestas únicas y por tanto sesgando los resultados. En efecto, la ausencia de otros evaluadores podría estar ignorando como respuestas o soluciones correctas otras que no fueron consideradas en este trabajo. Es por esto que queda planteada la posibilidad de profundizar este trabajo con la inclusión de múltiples evaluadores simultáneamente.

Desde una perspectiva didáctica, los resultados obtenidos sugieren que aquellos estudiantes que modelan mejor el problema son también quienes logran resolverlo de mejor forma. Los siguientes serían aspectos a tener en cuenta al momento de realizar la evaluación.

1. La explicación que los alumnos hacen de cómo se debe resolver el problema podría ser un predictor del éxito en la construcción de los programas; aspecto que solamente puede

ser ratificado, o no, en función de trabajos que profundicen esta línea de investigación y que analicen relaciones causales. Por tanto, para evaluar los conocimientos que tiene un alumno sobre programación, las preguntas asociadas a aspectos teóricos o el funcionamiento de las sentencias del lenguaje, podrían ser un indicador de menor calidad que las explicaciones sobre cómo resolver el problema.

2. La consigna usada para evaluar debería presentar un grado de abstracción lo suficientemente amplia como para permitir identificar lo que hemos denominado modelo. En tal sentido deberían evitarse los problemas con un alto grado de concretitud, donde la explicación del mismo sea una descripción de sentencias o el propio problema.
3. Podría considerarse negar a los alumnos el uso de una serie de palabras al momento de describir cómo resolver el problema. De este modo se podría matizar el punto anterior, a la vez que el alumno se ve enfrentado a un esfuerzo cognitivo significativo, relacionando nuevos conceptos y procedimientos que expliquen el problema.

Referencias

Aho, A. (2012). Computation and Computational Thinking. The Computer Journal, 55(7) (pp. 832–835). https://doi.org/10.1093/comjnl/bxs07. [GS Search]

Allen, L., Snow, E. y McNamara, D. (2015). Are you reading my mind?: modeling students' reading comprehension skills with natural language processing techniques. In Proceedings of the Fifth International Conference on Learning Analytics And Knowledge - LAK '15. https://doi.org/10.1145/2723576.2723617. [GS Search]

Bouvier, D., Lovellette, E., Matta, J., Alshaigy, B., Becker, B., Craig, M., Jackova, J.,… y Zarb, M. (2016). Novice programmers and the problem description effect. In Proceedings of the 2016 ITiCSE Working Group Reports (pp. 103-118). https://doi.org/10.1145/3024906.3024912 [GS Search]

Brennan, K. y Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In Proceedings of the 2012 annual meeting of the American educational research association (pp. 1-25). Disponible en [Link]. [GS Search]

Chomsky, W. (1957). Hebrew: The eternal language. Jewish Publication Society. [GS Search]

Clements, D. y Gullo, D. (1984). Effects of computer programming on young children's cognition. Journal of Educational Psychology, 76(6) (pp. 1051–1058). https://doi.org/10.1037/0022-0663.76.6.1051. [GS Search]

Crossley, S., Kim, M., Allen, L. y McNamara, D. (2019). Automated Summarization Evaluation (ASE) Using Natural Language Processing Tools. Artificial Intelligence in Education (pp. 84–95). https://doi.org/10.1007/978-3-030-23204-7. [GS Search]

Ebrahimi, A. (1994). Novice programmer errors: Language constructs and plan composition. International Journal of Human-Computer Studies, 41(4) (pp. 457-480). https://doi.org/10.1006/ijhc.1994.1069. [GS Search]

Fedorenko, E., Ivanova, A., Dhamala, R. y Bers, M. (2019). The Language of Programming: A Cognitive Perspective. Trends in Cognitive Sciences, 23(7) (pp. 525-528). https://doi.org/10.1016/j.tics.2019.04.010. [GS Search]

Fernández, L., Peña, R., Nava, F., y Velázquez, A. (2002). Análisis de las propuestas de la enseñanza de la programación orientada a objetos en los primeros cursos. En Actas de las VIII Jornadas de Enseñanza Universitaria de la Informática (JENUI'02). Universidad de Extremadura (pp. 433-440). [GS Search]

Fitzgerald, S., Simon, B. y Thomas, L. (2002). Strategies that students use to trace code: an analysis based in grounded theory. In proceedings of the first international workshop on Computing education research (pp. 69-80). https://doi.org/10.1145/1089786.1089793. [GS Search]

Frawley, W. (2013). Linguistic semantics. Routledge. [GS Search]

Grandell, L., Peltomäki, M., Back, R. y Salakoski, T. (2006). Why complicate things? Introducing programming in high school using Python. In Proceedings of the 8th Australasian. Conference on Computing Education. Disponible en https://crpit.scem.westernsydney.edu.au/. [GS Search]

Henklain, M., Carvalho, L., Feitosa, E. (2025). Dificuldades no aprendizado de programação: Um exame baseado em mapeamento sistemático da literatura e interpretação analítico-comportamental. Revista Brasileira de Informática na Educação, [S. l.], 33 (pp. 1484–1521). https://doi.org/10.5753/rbie.2025.5272. [GS Search]

Johnson-Laird, P. (2010). Mental models and human reasoning. Proceedings of the National Academy of Sciences, 107(43) (pp. 18243-18250). https://doi.org/10.1073/pnas.1012933107. [GS Search]

Kazemitabaar, M., Huang, O., Suh, S., Henley, A. Z., & Grossman, T. (2025). Exploring the design space of cognitive engagement techniques with ai-generated code for enhanced learning. In Proceedings of the 30th International Conference on Intelligent User Interfaces (pp. 695-714). https://doi.org/10.1145/3708359.3712104. [GS Search]

Kintsch, W. y van Dijk, T. (1978). Toward a model of text comprehension and production. Psychological Review, 85(5) (pp. 363–394). https://doi.org/10.1037/0033-295X.85.5.363. [GS Search]

Kintsch, W. (1998). The Representation of Knowledge in Minds and Machines. International Journal of Psychology, 33(6) (pp. 411-420). https://doi.org/10.1080/002075998400169. [GS Search]

Koulouri, T., Lauria, S. y Macredie, R. (2014). Teaching Introductory Programming: A Quantitative Evaluation of Different Approaches. ACM Transactions on Computing Education, 14(4) (pp. 1–28). https://doi.org/10.1145/2662412. [GS Search]

Lahtinen, E., Ala-Mutka, K. y Järvinen, H. (2005). A study of the difficulties of novice programmers. ACM SIGCSE Bulletin, 37(3) (pp. 14-8). https://doi.org/10.1145/1151954.1067453. [GS Search]

Liao, Y. y Bright, G. (1991). Effects of computer programming on cognitive outcomes: A meta-analysis. Journal of educational computing research, 7(3) (pp. 251-268), https://doi.org/10.2190/e53g-hh8k-ajrr-k69m. [GS Search]

Litwin, E. (1997). Las Configuraciones Didácticas. Paidós. [GS Search]

López-García, A., Urquiza-Fuentes, J., Mendes, A. J. y Caeiro-Rodríguez, M. (2025). Improving University Students' Learning of Programming Using Customised Pseudocode. Computer Applications in Engineering Education, 33(4). https://doi.org/10.1002/cae.70061. [GS Search]

Lye, S. y Koh, J. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? Computers in Human Behavior, 41 (pp. 51–61). https://doi.org/10.1016/j.chb.2014.09.012. [GS Search]

Maloney, J., Peppler, K., Kafai, Y., Resnick. M. y Rusk, N. (2008). Programming by choice: urban youth learning programming with scratch. In Proceedings of the 39th SIGCSE technical symposium on Computer science education (pp. 367-371). https://doi.org/10.1145/1352135.1352260. [GS Search]

Maloney, J., Resnick, M., Rusk, N., Silverman, B. y Eastmond, E. (2010). The Scratch programming language and environment. ACM Transactions on Computing Education (TOCE), 10(4), (pp. 1-15). https://doi.org/10.1145/1868358.1868363. [GS Search]

Mazumder, S. y Pérez, M. (2024). The Correctness of the Mental Model of Arrays After Instruction for CS1 Students. In Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (pp. 806-811). https://doi.org/10.1145/3626252.3630943. [GS Search]

McNamara, D. (2004). SERT: Self-Explanation Reading Training. Discourse Processes, 38(1) (pp. 1-30). https://doi.org/10.1207/s15326950dp3801_1. [GS Search]

Meerbaum-Salant, O., Armoni, M. y Ben-Ari, M. (2013). Learning computer science concepts with scratch. Computer Science Education, 23(3) (pp. 239-264). https://doi.org/10.1145/1839594.1839607. [GS Search]

Mikolov, T., Chen, K., Corrado, G. y Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv, Ithaca, Nueva York. https://doi.org/10.48550/arXiv.1301.3781. [GS Search]

Miños Fayad, A. (2016a). Uso didáctico de estrategias inductivas en un curso introductorio de programación estructurada. Tecné, episteme y didaxi, 39 (pp. 95-110). https://doi.org/10.17227/01203916.4583. [GS Search]

Miños Fayad, A. (2016b). Del decímelo con tus palabras al no usar las palabras... Una experiencia de evaluación desde la didáctica de la informática. InterCambios. Dilemas y Transiciones De La Educación Superior, 3(2) (pp. 77-81). [GS Search]

Miños Fayad, A. (2017). Elementos estructurantes de la Didáctica de la Informática. Virtualidad, Educación y Ciencia, 8(14) (pp. 100-110). Disponible en [Link]. [GS Search]

Mota, N., Furtado, R., Maia, P., Copelli, M. y Ribeiro, S. (2014). Graph analysis of dream reports is especially informative about psychosis. Scientific reports, 4(3691). https://doi.org/10.1038/srep03691. [GS Search]

Pesce, F. (2014). La didáctica en la formación de docentes para la enseñanza media en Uruguay. InterCambios. Dilemas y Transiciones De La Educación Superior, 1(1) (pp. 52-61). [Link]. [GS Search]

Piccolo, S. R., Denny, P., Luxton-Reilly, A., Payne, S. H., y Ridge, P. G. (2023). Evaluating a large language model's ability to solve programming exercises from an introductory bioinformatics course. PLOS Computational Biology, 19(9). https://doi.org/10.1371/journal.pcbi.1011511. [GS Search]

Prat, CS, Madhyastha, TM, Mo ttarella, MJ y Kuo, C. (2020). Relating natural language aptitude to individual differences in learning programming languages. Scientific reports, 10(1). https://doi.org/10.1038/s41598-020-60661-8. [GS Search]

Prather, J., Denny, P., Leinonen, J., Becker, B. A., Albluwi, I., Craig, Keuning, H.,… y Savelka, J. (2023). The robots are here: Navigating the generative ai revolution in computing education. In Proceedings of the 2023 Working Group Reports on Innovation and Technology in Computer Science Education (pp. 108-59). https://doi.org/10.1145/3623762.3633499. [GS Search]

Pressman, R., (2010). Ingeniería de software. Un enfoque práctico (7ª ed.). México: Mc Graw Hill Educación.

Renumol, V., Janakiram, D. y Jayaprakash, S. (2010). Identification of Cognitive Processes of Effective and Ineffective Students During Computer Programming. ACM Transactions on Computing Education, 10(3) (pp. 1–21). https://doi.org/10.1145/1821996.1821998. [GS Search]

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B. y Kafai, Y. (2009). Scratch: programming for all. Communications of the ACM, 52(11) (pp. 60-67). https://doi.org/10.1145/1592761.1592779. [GS Search]

Robins, A., Rountree, J. y Rountree, N. (2003). Learning and Teaching Programming: A Review and Discussion. Computer Science Education, 13(2) (pp. 137–172). https://doi.org/10.1076/csed.13.2.137.14200. [GS Search]

Sanders, I., Galpin, V. y Götschi, T. (2006). Mental models of recursion revisited. ACM SIGCSE Bulletin, 38(3) (pp. 138-142). https://doi.org/10.1145/1140124.1140162. [GS Search]

Scaffidi, C. y Chambers, C. (2012). Skill progression demonstrated by users in the Scratch animation environment. International Journal of Human-Computer Interaction, 28(6) (pp. 383-398). https://doi.org/10.1080/10447318.2011.595621. [GS Search]

Sentance, S., y Waite, J. (2021). Teachers' perspectives on talk in the programming classroom: Language as a mediator. In Proceedings of the 17th ACM Conference on International Computing Education Research (pp. 266-280). https://doi.org/10.1145/3446871.3469751. [GS Search]

Shute, V., Sun, C. y Asbell-Clarke, J. (2017). Demystifying computational thinking. Educational Research Review, 22 (pp. 142–158). https://doi.org/10.1016/j.edurev.2017.09.003. [GS Search]

Skalicky, S., Crossley, S., McNamara, D. y Muldner, K. (2017). Identifying Creativity During Problem Solving Using Linguistic Features. Creativity Research Journal, 29(4) (pp. 343-353). https://doi.org/10.1080/10400419.2017.1376490. [GS Search]

Soloway, E. (1986). Learning to program= learning to construct mechanisms and explanations. Communications of the ACM, 29(9) (pp. 850-858). https://doi.org/10.1145/6592.6594. [GS Search]

Valle-Lisboa. J. y Mizraji, E. (2007). The uncovering of hidden structures by Latent Semantic. Analysis. Information Sciences, 177(19) (pp. 4122–4147). https://doi.org/10.1016/j.ins.2007.04.007. [GS Search]

Valle-Lisboa, J., Pomi, A., Cabana, A., Elvevåg, B. y Mizraji, E. (2014). A modular approach to language production: Models and facts. Cortex, 55 (pp. 61–76). https://doi.org/10.1016/j.cortex.2013.02.005. [GS Search]

Van Dijk, T. (1980). Estructuras y funciones del discurso: una introducción interdisciplinaria a la lingüística del texto ya los estudios del discurso. Siglo XXI. [GS Search]

Van Dijk, T. (2019). El discurso como interacción social. Gedisa. [GS Search]

Villalobos, J. y Calderón, N. (2009). Proyecto Cupi2: un enfoque multidimensional frente al problema de enseñar y aprender a programar. Revista De Investigaciones UNAD, 8(2) (pp. 45-64). https://doi.org/10.22490/25391887.635. [GS Search]

Webb, M., Davis, N., Bell, T., Katz, Y., Reynolds, N., Chambers, D. y Sysło, M. (2017). Computer science in K-12 school curricula of the 2lst century: Why, what and when?. Education and Information Technologies, 22(2) (pp. 445-468). https://doi.org/10.1007/s10639-016-9493-x. [GS Search]

Wiedenbeck, S. y Ramalingam, V. (1999). Novice comprehension of small programs written in the procedural and object-oriented styles. International Journal of Human-Computer Studies, 51(1) (pp. 71–87). https://doi.org/10.1006/ijhc.1999.0269. [GS Search]

Wing, J. (2006). Computational thinking. Communications of the ACM. 49(3) (pp. 33-35). https://doi.org/10.1145/1118178.1118215. [GS Search]

Winslow, L. (1996). Programming pedagogy – A psychological overview. SIGCSE Bulletin, 28 (pp. 17-22). https://doi.org/10.1145/234867.234872. [GS Search]

Xie, B., Nelson, G. y Ko, A. (2018). An Explicit Strategy to Scaffold Novice Program Tracing. In Proceedings of the 49th ACM Technical Symposium on Computer Science Education - SIGCSE '18 (pp. 344-349). https://doi.org/10.1145/3159450.3159527. [GS Search]

Xie, B., Loksa, D., Nelson, G., Davidson, M., Dong, D., Kwik, H., Hui Tan, A., Hwa, L., Li., M. y Ko, A. (2019). A theory of instruction for introductory programming skills. Computer Science Education, 29(3) (pp. 205-253). https://doi.org/10.1080/08993408.2019.1565235. [GS Search]