Trabajo Damas

Gladys Cerro – Juan Pablo Sepulveda – Neftali Madariaga

Introducción

En este trabajo sobre el proyecto del juego de damas, deseamos llamar la atención sobre los distintos algoritmos que existen para desarrollar dicho juego, debido a que en la industria del software existen diversas soluciones que permiten desarrollar este juego. Para ello, tomamos 3 alternativas de algoritmos, desarrollados en C y C++, para el juego de damas, realizamos varias jugadas en cada uno de ellos y definimos cuál es mejor, según nosotros, para efectos de enfrentarse al computador. Es importante dejar en claro, que para efectos de este trabajo, no desarrollamos un algoritmo propio, sino que evaluamos las alternativas y las analizamos según varios parámetros expuestos a continuación, y que según nuestro criterio, son importantes para cualquier aplicación de este estilo.

Recordemos que el juego de damas es un juego de mesa que se efectúa entre 2 contrincantes, en donde el objetivo principal es despojar al rival de sus fichas, u obligarlo a realizar movimientos que puedan materializarse en la pérdida de sus fichas, mediante movimientos diagonales acotados por un conjunto de reglas. Está compuesto por un tablero que puede ser de 64 o 100 casillas, pero para este trabajo se analizaron alternativas para tableros de 64 casillas, las cuales están alternadas entre colores blancos y negros. Cada jugador posee 12 fichas de un color específico, por lo general uno recibe el color blanco y el otro el color negro, los turnos son de manera alternada, es decir cada uno realiza un movimiento para dar paso al rival, y solo pueden hacer 1 movimiento por turno, hacia adelante y de manera diagonal. Una vez que un jugador logre llegar con una ficha al lugar del contrincante, esta ficha pasa a llamarse “reina”, la cual tiene la virtud de poder efectuar movimientos de retroceso, el que las fichas comunes no pueden hacer.

También queremos destacar que para este tipo de juegos en donde es fundamental la concentración, y su principal estilo de juego es en persona, las aplicaciones deben ser amigables en su interfaz, esto significa que le permitan al usuario poder reconocer rápidamente sus fichas, las del rival y el tablero, brindándole de manera rápida todo el contexto del juego para desarrollar su estrategia. Como veremos a continuación, entre nuestras alternativas estudiadas hay 2 que no cumplen con dicha condición y transforman el juego en un problema a la hora de identificar las posiciones. Pero no solo debemos considerar este factor a la hora de elegir un algoritmo que satisfaga el juego de damas, sino que también debemos analizar el comportamiento del algoritmo al momento de establecer como jugador rival al computador, ya que la dificultad del juego está en que tan efectivo es el algoritmo al momento de tomar la decisión correcta en el movimiento de fichas. Algunos algoritmos siguen patrones, por lo cual basta con un par de jugadas para comprender dicho patrón y establecer una estrategia que permita superar a la máquina.

Desarrollo

Para la realización de este proyecto tuvimos que investigar sobre varios algoritmos que se utilizan para la resolución de un juego de damas, pero siempre teniendo en cuenta las restricciones planteadas. Para ello utilizamos varios algoritmos desarrollados en diversos lenguajes de programación, en donde analizamos su funcionamiento, su implementación y los recursos utilizados por el algoritmo para decidir por alguno.

La implementación de los algoritmos se realizaron en el siguiente entorno de hardware y software:

1. Hardware: Procesador Intel Core i5 @ 2.40 GHz, 4 Gb de memoria RAM, 500 Gb de disco duro.
2. Software: Sistema Operativo Ubuntu Linux 14.04 de 64 bits en donde el compilador utilizado es gcc.

1. Algoritmo Damas en C (Usuario vs Usuario)

El primer algoritmo analizado fue uno desarrollado en C, el cual mediante una matriz permite a los usuarios a través de un sistema de turnos alternado, realizar el juego mediante el clásico movimiento de fichas rojas y blancas. El usuario debe seleccionar la ficha a mover mediante el ingreso de sus coordenadas (fila y columna) para luego indicar el tipo de movimiento (izquierda o derecha). Una vez ejecutado el movimiento le corresponde al otro usuario realizar su turno mediante las mismas instrucciones para así generar un flujo de turnos que componen el juego. Lo primero que nos llamó la atención del algoritmo es que está pensado para 2 jugadores, por lo cual fue descartado de un principio ya que no cumplía con la condición inicial de ser un juego entre el usuario y el computador. Aún así lo analizaremos como un acercamiento a los demás algoritmos y como introducción a un desarrollo al juego de damas.

A continuación describiremos los mecanismos de toma de decisiones que utiliza el software para el llenado del tablero y para el movimiento de las fichas.

La función para el llenado del tablero es una que distribuye los 2 conjuntos de fichas, con la denominación de R para las rojas y B para las blancas, en un tablero de 8 casillas por 8 casillas (64 casillas en total) pintadas alternativamente. Para ubicar cada ficha en su casilla correspondiente utiliza 2 funciones for en donde la primera llena hace el llenado por filas y la segunda, que se encuentra anidada, realiza el llenado por columnas. Así permite consultar posición por posición si esta cumple con la condición de recibir la ficha indicada. Las rojas deben estar en las 3 primeras filas (de 0 a 2), cuya posición es distribuida alternativa asumiendo la posición 0,0 como cuadrado rojo, es por esto que realiza la suma de las posiciones y verifica que sea de tipo par, para así disponer cada ficha. De manera contraria a la ubicación de las filas para las fichas rojas, es decir en las últimas 3 filas (de 5 a 7) se ubicarán las blancas, las cuales cumplen la misma condición que las rojas en que la suma de las posiciones debe ser de tipo par. El resto de las casillas que no cumplan dicha condición serán vacías para que las fichas puedan moverse en la totalidad del tablero.

Ahora la función del movimiento de las piezas está establecida por el siguiente criterio que establecemos a continuación. Lo primero que verifica esta función de movimiento de la ficha es que exista una del color a la que corresponde el turno en la posición ingresada por el usuario, de no ser así despliega un mensaje al usuario indicando el error. En caso de que exista la ficha seleccionada la función realiza una serie de validaciones, partiendo por establecer que el movimiento de la ficha

seleccionada no se desplace hacia fuera de las dimensiones del tablero, en caso de que así sea se despliega un mensaje notificando al usuario de su error. Si el movimiento se encuentra dentro de las dimensiones del tablero, este debe validar como segundo requisito que el movimiento se ejecute de manera diagonal izquierda o diagonal derecha, utilizando para ello la suma de la primera coordenada y la resta de la segunda según corresponda; así se asegura de que el movimiento se realice diagonalmente. Como tercer requisito se debe validar que el movimiento deseado sea aplicable a una casilla vacía, de tal manera que la función verifica que en esa nueva coordenada exista el número 0 (la denominación de casilla vacía) y no esté ocupado por una ficha del mismo color; en caso de que se cumpla el requisito la ficha se desplaza hacia la nueva posición. Como tercer requisito la función debe validar que exista una ficha del color opuesto, para ello verifica si en la nueva coordenada existe la letra del color opuesto, y de ser así, al movimiento indicado por el jugador que genera la nueva coordenada se le aumenta en 1 a la abscisa y se aumenta o disminuye en 1 a la ordenada (dependiendo del color de la ficha).

La interfaz de este algoritmo es bastante deficiente, ya que no permite diferenciar con claridad las piezas de las casillas en blanco, logrando que el usuario se preocupe más de identificar sus piezas y no de la estrategia a implementar. Como mencionamos en la introducción es muy importante que las aplicaciones de juegos de mesa tengan una buena interfaz, ya que el usuario debe utilizar más tiempo en planear una estrategia basada en movimientos eficientes con el fin de ganar la partida y no intentar identificar el contexto en el cual se está desarrollando el juego.

El rendimiento de este algoritmo es bastante pobre, debido a que el tiempo que inviertan ambos jugadores es demasiado extenso producto del constante requerimiento de solicitar el ingreso de coordenadas exactas, sumado a la pobre interfaz gráfica, hace aún más tedioso el juego.

2. Algoritmo Damas en C++ (Usuario vs Computador)

Este algoritmo está desarrollado para el juego de damas en lenguaje C++, el cual es muy parecido al anterior, pero esta vez sí permite competir contra el computador siguiendo los mismos criterios que el anterior establecido para 2 jugadores. Esta aplicación, al igual que el anterior, llena una matriz de 8x8 que representa al tablero (cuyas filas y columnas van de 1 a 8), con la diferencia en que denomina a las fichas del computador con el carácter “q” y a las del usuario con la letra “h”. El llenado del tablero se hace ubicando las fichas correspondientes al computador entre las filas 6 y 8, y las del usuario entre las filas 1 y 3, entregando siempre al computador la posibilidad de comenzar el juego. La función que realiza dicha acción evalúa los mismos criterios que el algoritmo anterior, estableciendo las fichas en las casillas cuya suma de coordenadas sea de tipo par, cumpliendo de esta manera el requerimiento propio al juego de damas. A diferencia del anterior es que la matriz correspondiente al tablero no se llena completamente, sino que solo ubica las fichas dejando en blanco las demás casillas, lo que puede dificultar al usuario analizar el tablero completo.

Para este algoritmo los movimientos de las fichas están definidos en 2 funciones, una representa los criterios a evaluar los movimientos del usuario y la otra realiza los movimientos del computador, en donde esta última función contiene menos restricciones ya que está desarrollado para que el computador no haga movimientos no permitidos, como sí lo haría el usuario. La primera de ellas se llama **int Board::trymove** y corresponde a los movimientos que puede ejecutar el computador.
En el caso de los movimientos del usuario, la función encargada de realizar dichos movimientos se llama **human_move**, la cual solicita al usuario que ingrese las coordenadas de la pieza a mover (número de fila y columna), para luego solicitar el ingreso de la nueva casilla a mover la ficha. A

diferencia del algoritmo anterior, el cual preguntaba al usuario que indique el tipo de movimiento, ya sea derecha o izquierda, y el algoritmo solo colocaba la pieza en la casilla de destino según la función matemática por la que se rigen los movimientos de las damas, este algoritmo como recién mencionamos, primero solicita al usuario ingresar la casilla de destino de la ficha a mover para luego validar si es posible realizar dicho movimiento mediante la fórmula matemática (aumenta en 1 el valor de la abscisa y si es un movimiento a la derecha también aumenta en 1 el valor de la ordenada, en caso de ser a la izquierda disminuye en 1 el valor).

La interfaz de este algoritmo es mejor que la utilizada por la alternativa 1 debido a que las casillas no utilizadas por las fichas si están en blanco, es decir no están identificadas mediante algún carácter que pueda confundir al usuario en la diferenciación entre sus fichas, las del computador y las casillas del tablero. Ahora, el problema de interfaz que presenta esta alternativa es que los caracteres no están bien alineados en la matriz y puede tender a la confusión en ese aspecto.

En este caso, el rendimiento de este algoritmo es mucho mejor que el anterior, ya que al ser el usuario el que juega contra el computador, el tiempo de juego depende exclusivamente de las habilidades del usuario, puesto que el computador siempre desarrollará movimientos rápidos pero no siempre los mejores.

3. Algoritmo Damas en C++ (Usuario vs Computador)

Este algoritmo, al igual que la alternativa 2 está desarrollado para el juego de Damas en C++, en donde se enfrenta el usuario contra el computador, siguiendo las reglas internacionales de este juego. El llenado del tablero es igual que todos los demás, en donde se genera una matriz de 8x8 con casillas alternativas entre blancas y negras, ubicando las fichas de ambos rivales según los colores. El criterio de decisión está establecido por el movimiento de las fichas en las casillas, y al igual que en los otros 2 juegos, la suma de coordenadas sea de tipo par, cumpliendo de esta manera el requerimiento propio al juego de damas.

Esta alternativa nos pareció mucho más elaborada que las anteriores, ya que tiene un desarrollo más profundo en cuanto a las instrucciones a seguir. Por ejemplo, para ejercer los movimientos del usuario no lo hace solicitando el ingreso de coordenadas ni tampoco preguntando el tipo de movimiento, sino que el usuario puede desplazar con los clicks del mouse el movimiento de las fichas simulando tal cual es en la vida real. Para la validación de los movimientos el usuario primero debe seleccionar la ficha a mover para que el algoritmo sepa la coordenada exacta de la ficha que será movida, y luego dar paso a la selección de la casilla de destino, en donde el algoritmo se encarga de validar si el movimiento es permitido mediante la suma de coordenadas ya expuestas en las alternativas anteriores. Si un movimiento no es válido la ficha simplemente no se desplazará, haciendo una gran diferencia entre la alternativa anterior en donde el usuario por cada mal movimiento pierde un turno, factor no menor ya que el juego real no se caracteriza por pérdidas de turnos.

La interfaz es mucho más amigable, y es por esto es que mencionamos que el desarrollo es mucho más elaborado que las alternativas anteriores, ya que busca una aproximación mucho más cercana al juego real. Como mencionamos en la introducción este punto es muy importante para un juego de este estilo, ya que lo hace mucho más ágil y fácil de jugar, liberando al usuario de una carga extra por tratar de entender el contexto del juego.

Conclusiones

Durante el desarrollo de este trabajo, nos encontramos con una cantidad de algoritmos para el desarrollo del juego de damas bastante importante, en donde tuvimos que ir analizando las distintas alternativas propuestas y poder ver cual es la mejor para implementar este juego.

Como primer aspecto nos enfocamos en la interfaz de los algoritmos, ya que como hemos mencionado bastante es un punto que no puede ser tratado al lote. No podemos entregarle a un usuario una aplicación de un juego de mesa que tenga una interfaz complicada, que lo haga utilizar tiempo y concentración en identificar las piezas y las casillas del tablero, sino que debemos entregarle de la forma más amigable posible el contexto en el cual se desarrolla el juego.

Otro punto muy importante para nosotros es que exista una coherencia y fluides en los turnos, es decir que el usuario no pierda un turno producto de un movimiento inválido. Muchos algoritmos utilizan este criterio en donde al momento de validar los movimientos y estos no llegan a estar permitidos (por ejemplo intentar mover una ficha fuera del tablero) dan paso a la finalización de dicho turno entregando la posibilidad de jugar al computador. Este punto es relevante debido a que si el jugador se equivoca en la totalidad de sus movimientos terminará jugando el computador prácticamente solo, lo que no corresponde al juego real de damas.

Ahora mencionaremos tal vez el punto que más marca la diferencia entre un algoritmo y el otro, y es que el criterio de decisión de las fichas para los turnos del computador. Durante un estudio de las distintas variables utilizadas en el juego de damas descubrimos que existen algoritmos especiales para este tipo de juego, en donde el principal se llama **Minimax** y consiste en ser una búsqueda limitada en profundidad, siendo la estrategia la que prevalece para
búsquedas en árboles de juego. Consiste en buscar hacia abajo hasta cierta profundidad y trata a
los nodos de dicha profundidad como si fueran nodos terminales. Minimax tiene por objetivo devolver el mejor movimiento de todos los posibles según la profundidad explorada, y una vez determinado dicho movimiento será el que se ejecute; para el caso de este juego, el movimiento es el índice de la casilla. Idealmente Minimax debe expandir todo el árbol de juego y llegar hasta cada nodo, pero debido a que su complejidad es de tipo exponencial es casi imposible implementarlo en la práctica. Debido a esto es que se hacen optimizaciones como podas (Poda Alfa – Beta es un ejemplo), y el algoritmo debe aplicar alguna técnica como backtracking. También Minimax invoca una función
heurística denominada función de evaluación. La dificultad del juego está en qué tan bien devuelve el mejor movimiento, y para ello por lo general se le asignan puntajes a las casillas de acuerdo a una heurística, por ejemplo considerando la proporción de fichas blancas y negras.

Nosotros para inclinarnos por una de las alternativas ponderamos todos los puntos antes mencionados y concluimos cual satisfacía la mayoría de ellos.

La primera alternativa la descartamos porque no cumplía el requerimiento básico de que enfrentara al computador, por lo tanto su algoritmo era bastante básico, ya que solo validaba que los movimientos ingresados por ambos usuarios fuese permitido. También lo descartamos porque su interfaz era muy complicada para el usuario, durante la cantidad de partidas que realizamos nos generaba una confusión en tratar de comprender el contexto en el cual se estpa desarrollando el juego. Sin embargo nos sirvió mucho como una aproximación al juego de damas para entender el funcionamiento de los algoritmos y los mecanismos que se utilizan para la toma de decisiones, algo que uno hace de manera simple en la vida real es mucho más complicado programarlo en una aplicación.

La segunda alternativa cumplía el requerimiento de enfrentar al usuario contra el computador, y la interfaz era mucho más amigable que la alternativa anterior, pero el punto que nos hizo descartarla fue debido a que al realizar un movimiento inválido el usuario perdía su turno. Esto fue bastante desilusionante, ya que el juego original no prohíbe al jugador de su turno, sino que le permite jugar hasta conseguir un movimiento válido, indicando ahí que la finalización del turno ha llegado. También lo que nos hizo buscar una tercera alternativa fue que la interfaz tampoco era de las mejores y los mecanismos de ingreso de las coordenadas era bastante engorrosa. Creemos que un buen juego de damas debe ser apto para todo tipo de público que solo debe tener en conocimiento las reglas del juego, y no solicitar al usuario un ingreso de coordenadas para realizar el movimiento, lo que conlleva a manejar un vocabulario más técnico que escapa al reglamento del juego.

La tercera alternativa, al igual que la segunda, cumplía con el requerimiento de enfrentar al usuario con el computador, pero esta vez si que la interfaz estaba bastante más desarrollada, simulando en su totalidad el juego aplicado en la vida real. Se podían diferenciar bastante bien las piezas de los jugadores, las casillas del tablero, y el desplazamiento de fichas es mediante el uso del click del mouse. Esto marca una diferencia fundamental entre las otras dos soluciones ya antes expuestas, debido a que el jugador puede realizar sus movimientos tal cual como lo haría en un tablero físico, sin perder turno frente a un mal movimiento y sin ingresar coordenadas para la realización de movimientos. También el algoritmo es mucho más efectivo que los demás debido a que los movimientos del computador siguen la heurística ya antes mencionada entregando entre las 3 alternativas el mejor movimiento, aumentando la dificultad para el usuario.

Finalmente podemos concluir que la elaboración de este trabajo fue fundamental para darnos una perspectiva real sobre lo que es el desarrollo de aplicaciones para esta materia, en donde muchas veces simular el pensamiento humano es demasiado complicado para el computador debido a la cantidad de variables a influir en una toma de decisiones; como mencionamos anteriormente lo que hacemos trivial.