Búsqueda por Similitud de Posiciones de Ajedrez

Diego Gonzalez[1], Andrés Pascal[1], Anabella De Battista[1],
Norma Herrera[2]

[1] Dpto. de Sistemas de Información, Universidad Tecnológica Nacional,
Entre Ríos, Argentina,
{gonzalezdiego014, andrespascal22, anadebattista}@gmail.com
[2] Dpto. de Informática, Universidad Nacional de San Luis, Argentina,
nherrera@unsl.edu.ar

Resumen. Las búsquedas por similitud constituyen un c ampo de estudio de gran importancia en la actualidad. En el presente trabajo se propone una función distancia para consultar por similitud posiciones de ajedrez sobre bases de datos de partidas, ya que actualmente estas consultas están limitadas a búsquedas exactas. Se evalúa su comportamiento mediante distintos tipos de pruebas sobre las fases del juego.

Palabras clave: Búsquedas por similitud. Ajedrez. Espacios métricos. Función de distancia

1 Introducción

Las búsquedas en las bases de datos tradicionales se basan en el concepto de búsqueda exacta: la base de datos es dividida en registros, teniendo cada registro campos completamente comparables; una consulta a l a base retorna todos aquellos registros cuyos campos coincidan con los aportados en la búsqueda. En la actualidad, muchas aplicaciones tienen como necesidad buscar en grandes bases de datos objetos que sean similares a uno dado. Este tipo de búsqueda se conoce con el nombre de *búsqueda por similitud* y surge en diversas áreas [1, 2]. En este artículo nos enfocamos en la búsqueda por similitud de posiciones en grandes bases de datos de partidas de ajedrez. Denominamos *posición* a un tablero de ajedrez con piezas ubicadas en celdas definidas, resultante de 0 o más movimientos de una partida.

El juego de ajedrez es foco de estudio desde hace más de cinco décadas en varias disciplinas, por ejemplo, podemos nombrar entre ellas: inteligencia artificial, elaborando algoritmos para la evaluación de las posiciones [3] y ciencias educacionales realizando estudios que exponen las ventajas de su aprendizaje para el desarrollo de habilidades personales [4].

Dentro de las ciencias de la computación, la mayor parte de los estudios se basan en el análisis de variantes sobre una posición o los llamados motores de evaluación de posiciones. Con respecto a las bases de datos, se han realizado estudios para obtener

conocimiento sobre la comparación de distintos motores como así también para la obtención de conocimiento sobre las partidas [5, 6]. Sobre las aplicaciones que consultan partidas, hoy en día se utilizan búsquedas por exactitud de las posiciones; es decir que las posiciones que se comparan deben tener cada pieza ubicada en el mismo lugar. Éste tipo de búsquedas es limitado ya que, en el ajedrez profesional, los jugadores no memorizan la exacta posición de las piezas, sino que identifican patrones dentro de una posición [7].

Aún no existe un sistema que permita búsquedas por similitud sobre posiciones de Ajedrez. La idea de las búsquedas por similitud, es obtener posiciones que puedan tener algunas variaciones respecto a l a posición consultada. En éste artículo proponemos una función de distancia para las búsquedas por similitud de posiciones de ajedrez sobre una base de datos de partidas. Este tipo de consulta es útil para que aprendices del juego, ante una situación determinada, conozcan posibles jugadas o estrategias que siguieron jugadores profesionales en partidas reales. También podría ser utilizada por motores de ajedrez para podar los árboles de búsqueda o pa ra mejorar su capacidad de juego.

Éste artículo está organizado de la siguiente manera: en la Sección 2 se describe el trabajo relacionado. En la Sección 3 se define y analiza la función distancia desarrollada, mostrando resultados experimentales de su funcionamiento en la Sección 4. Finalmente en la Sección 5 se expresan las conclusiones y el trabajo futuro.

2 Trabajo Relacionado

2.1 Búsquedas en Espacios Métricos

El problema de búsqueda de posiciones de ajedrez puede abstraerse como un universo de objetos U (las posiciones) y una función de distancia d que modela la similitud entre los objetos del universo. El par (U, d) se denomina *espacio métrico* y es uno de los modelos más utilizados como soporte para las búsquedas por similitud [1, 2].

Formalmente un espacio métrico es un par (U, d) donde U es un universo de objetos y $d : U \times U \rightarrow R+$ es una función de distancia definida entre los elementos de U que mide la similitud entre ellos; esto significa que a menor distancia, más cercanos o similares son los objetos. Esta función d cumple con las propiedades características de una función métrica:

(a) $\forall\, x, y \in U, d(x, y) \geq 0$ (positividad)
(b) $\forall\, x, y \in U, d(x, y) = d(y; x)$ (simetría)
(c) $\forall\, x \in U, d(x, x) = 0$ (reflexividad)
(d) $\forall\, x, y, z \in U, d(x, y) \leq d(x, z) + d(z, y)$ (desigualdad triangular)

La base de datos es un subconjunto finito $X \subseteq U$ de cardinalidad n. Existen dos tipos básicos de consultas en este modelo, la *búsqueda por rango* y la *búsqueda de los k vecinos más cercanos*.

La búsqueda por rango se denota $(q, r)_d$, donde $q \in U$ es el elemento de consulta que llamaremos *query,* y r es el radio de tolerancia. Una búsqueda por rango consiste en recuperar todos los objetos de la base de datos que se encuentren como máximo a distancia r de q, es decir:

$$(q, r)_d = \{x \in X / d(q, x) \leq r \}$$

La búsqueda de los k vecinos más cercanos se expresa como $NN_k (q)_d$, y consiste en recuperar los k elementos de la base de datos que posean menor distancia a q, es decir:

$$NN_k (q)_d = A, \ |A| = k, \ A=\{ x \in X / \ \forall y \in (X-A), d(q, x) \leq d(q, y) \}$$

Para resolver estas consultas con mayor eficiencia que $O(n)$ evaluaciones de la función de distancia (que es el costo de recorrer secuencialmente la base de datos), se utilizan índices que ahorran cálculos durante el proceso de búsqueda [8, 9, 10].

2.2 Búsqueda de posiciones de Ajedrez

Particularmente dentro de la inteligencia artificial nos encontramos con varios trabajos relacionados con algoritmos para la evaluación de posiciones, en los que se aplican funciones de evaluación cuyo objetivo principal es obtener variantes[1] que proporcionen beneficios al jugador que tiene su turno en un juego [3].

Existen hoy en día diversos tipos de base de datos de ajedrez y con gran cantidad de datos, como por ejemplo los libros de Aperturas[2], en los cuales se aplican búsquedas exactas; es decir, se requiere que coincida la posición de cada pieza de la consulta con la misma pieza en el elemento consultado. A su vez, existen bases de datos sobre partidas jugadas, posiciones de medio juego y posiciones finales. Pero en todos los casos las consultas se realizan de la misma manera, lo cual es una limitante ya que cualquier detalle distinto sin relevancia no permite encontrar posiciones que serían de utilidad. En lugar de ello, hay propuestas como en [11] que muestran la importancia de que la búsqueda tenga en cuenta aspectos semánticos que incluyan el concepto o idea de la posición.

Por otro lado, en [7] se clasifica la idea de similitud en el ajedrez en similitud visual y similitud abstracta. Los Jugadores profesionales se centran principalmente en la similitud abstracta, identificando patrones dentro de cada posición, a diferencia de los jugadores principiantes que se basan en similitudes visuales.

Es importante considerar que la cantidad de combinaciones posibles dentro del juego está dada por la siguiente formula: $64! / (32!*(8!)^2*(2!)^6)$ [12] dando un resultado aproximado de 10^{43}. En este cálculo hay posiciones que son ilegales para el juego, por lo cual se estima que la cantidad de posiciones válidas es alrededor de 10^{40}.

1 Conjunto de jugadas.

2 Jugadas iniciales.

Al consultar una posición, es difícil definir todos los patrones que puede encontrar un profesional del juego. En su lugar, una solución alternativa es contar con una función que calcule la similitud entre dos posiciones.

3 Función Distancia Propuesta

Existen decenas de funciones de distancia genéricas, que pueden aplicarse a distintos problemas [13]. Sin embargo, en casos especiales es necesario definir una función específica que represente el significado de la similitud entre dos objetos para el problema particular. En el caso de las posiciones de Ajedrez, la función que se define en este artículo se basa en la idea de medir la cantidad de movimientos necesarios para transformar una posición en otra, y en caso de que no se pueda, incluir una penalización en el cálculo que represente la ausencia de la pieza.

3.1 Descripción de la Función de Distancia

Un tablero se compone de sesenta y cuatro casillas en las cuales se ubican treinta y dos piezas divididas en dos bandos de diferente color. Dentro de la base de datos tenemos conjuntos de posiciones que corresponden a partidas jugadas. Cada posición se representa con una notación ajedrecística llamada FEN[3]; dicha notación es una cadena de texto que describe la ubicación de cada una de las piezas representándola mediante una letra. Las letras en minúscula representan las piezas negras y las letras en mayúsculas las blancas. Se utilizan números para representar las casillas vacías y la barra de división como separación de cada fila. En la Fig. 1 se muestra la posición inicial de una partida y el FEN asociado.

Fig. 1. FEN asociado: *rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR*

El objetivo de la función distancia es obtener el costo mínimo en movimientos que resulta de posicionar cada una de las piezas de un tablero origen en la misma posición de un tablero destino. El costo del movimiento de cada pieza se calcula como si el tablero estuviese vacío. En caso de que no sea posible por falta de la misma pieza en el otro tablero, se introduce un costo que depende de tipo de pieza, pero que es siempre mayor al máximo costo de movimientos de dicha pieza. Los costos utilizados

3 Forsyth-Edward Notation

asociados a las piezas fueron: Dama = 10, Torre = 8, Alfil = 7, Caballo = 6 y Peón = 5. El Rey no posee costo ya que siempre debe existir en ambos bandos.

Cada tipo de pieza tiene una función para obtener la cantidad de movimientos, ya que las distintas piezas del juego tienen movimientos diferentes. Para las piezas Rey, Dama y Alfil, se validan los movimientos de cada pieza y en base al movimiento de cada una se devuelve el valor correspondiente. Para las restantes piezas: Torre, Caballo y Peón, al no ser piezas únicas (par de Torres, par de Caballos, ocho Peones), se construye una matriz de costos de movimientos y se utiliza el método húngaro [14] para determinar la asignación óptima de correspondencia.

3.2 Algoritmo

A continuación se presenta el pseudocódigo del algoritmo de la función de distancia propuesta.

```
Función DistanciaAjedrez (pos1, pos2)
  distancia:= 0
  Foreach color in [blanco, negro]
    inicializar(matrizPeones)   -- Peones
    Foreach peon1 in piezasDe(pos1, color)
      Foreach peon2 in piezasDe(pos2, color)
        matrizPeones[peon1, peon2]:=
            movs( Peon,
                  casillaDe(peon1,pos1, color),
                  casillaDe(peon2,pos2, color))
    distPeones:= metodoHungaro(matrizPeones)

    inicializar(matrizCaballos)   -- Caballos
    Foreach caballo1 in piezasDe(pos1, color)
      Foreach caballo2 in piezasDe(pos2, color)
          matrizCaballos:= movs( Caballo,
                          casillaDe(caballo1, pos1, color),
                          casillaDe(caballo2, pos2, color))
    distCaballos:= metodoHungaro(matrizCaballos)

    inicializar(matrizTorres) - Torres
    Foreach torre1 in piezasDe(pos1, color)
      Foreach torre2 in piezasDe(pos2, color)
        matrizTorres:= movs( Torre,
                           casillaDe(torre1, pos1,color),
                           casillaDe(torre2, pos2, color))
    distTorres:= metodoHungaro(matriztorres)
    -- Piezas únicas
    distRey:= movs( Rey, casillaDe(Rey, pos1, color),
```

```
                    casillaDe(Rey, pos2, color))
distDama:= movs( Dama, casillaDe(Dama, pos1, color),
                       casillaDe(Dama, pos2, color))
distAlfilBlanco:= movs(  AlfilBlanco,
                    casillaDe(AlfilBlanco, pos1, color),
                    casillaDe(AlfilBlanco, pos2, color))
distAlfilNegro:= movs( AlfilNegro,
                    casillaDe(AlfilNegro, pos1, color),
                    casillaDe(AlfilNegro, pos2, color))

distancia:= distancia + distRey + distDama +
            distCaballos + distPeones +distTorres +
            distAlfilBlanco + distAlfilNegr
Return distancia
```

La función *casillaDe(Pieza, Posición, Color)* devuelve la casilla (compuesta por fila y columna) correspondiente a la Pieza del Color dado, en el tablero representado por Posición. La función *movs(TipoDePieza, Casilla1, Casilla2)* devuelve la cantidad mínima de movimientos de una pieza del TipoDePieza, desde la Casilla1 a la Casilla2. Por problemas de espacio no se explica en detalle el mecanismo de cálculo de esta función.

3.3 Ejemplo

A continuación se presenta un ejemplo del cálculo de la Función de Distancia para un caso en que los tableros no contienen las mismas piezas. La Fig. 2 muestra las posiciones comparadas y Tabla 1 el cálculo de la función de distancia.

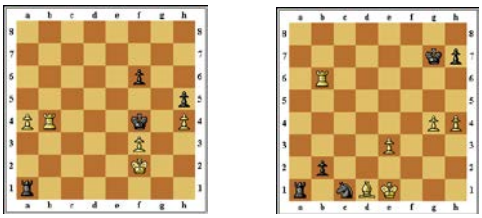

Fig. 2. Comparación de dos posiciones con distintas piezas

Tabla 1. Cálculo de la función de distancia para las posiciones de la **Fig. 2**.

Pieza	Color	Casilla inicial	Casilla objetivo	Distancia
Rey	Blanco	F2	E1	1
Torre	Blanco	B4	B6	1
Alfil	Blanco	-	D1	7
Peón	Blanco	A4	-	5

Peón	Blanco	F3	G4	1
Peón	Blanco	H4	H4	0
Rey	Negro	F4	G7	3
Torre	Negro	A1	A1	0
Caballo	Negro	-	C1	6
Peón	Negro	F6	B2	4
Peón	Negro	H5	H7	2
			Total	**30**

3.4 Análisis de la Función de Distancia como métrica

Para que una función sea métrica, debe de cumplir cuatro propiedades: positividad, reflexividad, simetría y desigualdad triangular.

La función propuesta cumple con la positividad, ya que es una sumatoria de números naturales (cantidad de movimientos de una pieza para alcanzar una posición a partir de otra). Es reflexiva, ya que cuando se mide la distancia entre dos posiciones iguales al encontrarse las piezas en los mismos lugares, el resultado siempre es 0. También es simétrica, debido a que la cantidad mínima de movimientos de una pieza para ir desde una casilla de origen a una casilla destino, es siempre la misma que la cantidad mínima desde la casilla destino a la de origen, y la función se calcula como la sumatoria de estas cantidades.

Respecto a la desigualdad triangular, aunque todavía no se hizo el estudio analítico de la función, se realizaron una serie de pruebas para verificar su cumplimiento estableciendo tres conjuntos a los cuales se les aplicó la función distancia entre sus elementos. En total se evaluaron 60.000 ternas y en todos los casos se verificó el cumplimiento de esta propiedad. A pesar de que esta verificación no constituye una demostración, es un punto de partida importante, ya que en caso de no cumplirla para una cantidad reducida de casos, igualmente se podrían utilizar índices métricos para acelerar la búsqueda tal como se plantea en [15].

4 Resultados Experimentales

Para la evaluación de la función distancia, se obtuvo de Internet una base de datos con un total de 179564 posiciones correspondientes a 2273 partidas. El objetivo de los experimentos fue verificar la eficacia o precisión de la función, es decir, que los objetos resultantes de una consulta sean los esperados. Para ello se definieron tres situaciones distintas; aperturas, medio juego y finales, y se obtuvieron lotes de consultas de dos maneras: consultas externas (posiciones extraídas de otras partidas de ajedrez que no están en la base de datos) y consultas de posiciones modificadas (en este caso se extraen posiciones de la base de datos a las cuales se les realizan modificaciones y el resultado se utiliza como consulta).

4.1 Consultas externas

4.1.1 Pruebas de Apertura

Para las pruebas de apertura se obtuvieron 40 posiciones desde Internet que se corresponden a las primeras 10 jugadas de una partida y con distinto código de Apertura perteneciente a la Enciclopedia de Aperturas de Ajedrez (ECO). La enciclopedia está dividida en cinco categorías que se representan con las letras desde A hasta E, y cada categoría incluye hasta cien subcategorías.

El objetivo de esta prueba fue determinar si el vecino más cercano tiene la misma categoría y subcategoría de apertura que la consulta.

Tabla 2. Pruebas externas de Apertura.

	Aciertos	**%**
Aperturas	40	100 %

En la Tabla 2 se muestran los resultados de las consultas externas de Aperturas. En el 100% de los casos la función clasificó las aperturas de manera correcta.

4.1.2 Pruebas de Medio Juego y Finales

Para las pruebas de Medio Juego y Finales se obtuvieron desde Internet 100 posiciones de Medio Juegos correspondientes a jugadas entre 15 y 20 movimientos de una partida, y 50 posiciones Finales correspondientes a jugadas mayores a 40 de una partida. Se analizaron los 10 vecinos más cercanos, verificando su similitud visualmente.

Tabla 3. Pruebas externas de Medio Juegos y Finales.

	Aciertos	**%**
Medio Juegos	848	84,80 %
Finales	393	78,60 %

En la Tabla 3, se muestran los resultados de las pruebas de Medio Juego y Finales. Podemos observar que para el Medio Juego la función devuelve una mayor cantidad de aciertos que en las posiciones de Finales. Esto es principalmente porque en los Finales de juego hay una menor cantidad de piezas que en los Medio Juegos, por lo que la existencia o falta de una pieza cobra mucha más relevancia y la función de distancia propuesta no tiene en cuenta este aspecto. Esta situación muestra que es necesario ajustar los valores de los pesos máximos de las piezas, ya que, por ejemplo,

con los valores actuales es lo mismo que falten dos peones (valor 5 cada uno) que una dama (valor 10).

4.2 Consultas de posiciones modificadas

Para las pruebas de modificación, se obtuvieron tres posiciones de la base de datos por cada fase del juego y se le realizaron modificaciones. Las modificaciones realizadas a los tableros de Apertura fueron de movimiento y eliminación de piezas, mientras que para las pruebas de Medio Juego y Finales, se realizaron modificaciones de movimiento, eliminación y agregado de piezas.

Se realizaron 5 modificaciones a cada posición de Apertura, 8 modificaciones a cada posición de Medio Juego y 6 modificaciones a cada posición Final. En este caso se obtuvo el vecino más cercano de cada consulta y se lo clasificó como Acierto si el elemento resultante pertenecía a la misma partida.

Tabla 4. Pruebas de posiciones modificadas de Apertura, Medio Juegos y Finales.

	Aciertos	**%**
Apertura	13	86,67%
Medio Juegos	23	95,83%
Finales	17	94,44%

Como podemos observar en la Tabla 4, para las pruebas de modificación de posiciones de la base de datos, los porcentajes de acierto son elevados, lo que indica que la función propuesta es representativa de la similitud real de las posiciones. El valor menor de aciertos es el de las Aperturas, ya que al inicio, las partidas de un mismo tipo son muy similares, por lo cual la función puede retornar un elemento de otra partida parecida.

5 Conclusiones y Trabajo Futuro

En este artículo proponemos una función distancia para búsquedas por similitud de posiciones de ajedrez en una base de datos de partidas jugadas y mostramos su buen comportamiento en una base de datos real. Esta función se puede utilizar para clasificar Aperturas, como soporte al aprendizaje del ajedrez y para mejorar el funcionamiento de motores de juego, entre otros usos.

La función se puede mejorar aún, para que en la etapa de final de un juego tenga en cuenta que la calidad y cantidad de piezas es mucho más importante que sus posiciones.

A continuación, se enumeran los trabajos que quedan por realizar:

- Analizar las posiciones finales y modificar la función distancia incluyendo la cantidad y valor relativo de las piezas, ya que a medida que el juego avanza, el valor de cada pieza aumenta.

- Ajustar el valor de la distancia de los peones cuando no correspondan en un movimiento válido.
- Demostrar que la función distancia sea métrica para posteriormente plantear soluciones en la eficiencia de las consultas.
- Realizar ajustes a la función distancia donde se tengan en cuenta los patrones de una posición para realizar búsquedas de forma abstracta.

Referencias

1. E. Chavez, G. Navarro, R. Baeza-Yates, and J.L. Marroquin. *Searching in metric spaces*. ACM Computing Surveys, 33(3): 273–321, (2001).
2. R. Baeza-Yates. Searching: an algorithmic tour. In A. Kent and J. Williams, editors, Encyclopedia of Computer Science and Technology, volume 37, pages 331-359. Marcel Dekker Inc. (1997).
3. Vazquez-Fernandez, E., Coello, C.A., Sagols , F .D.: An Evolutionary Algorithm for Tuning a Chess Evaluation Function: Applied Soft Computing, vol. 13, pp. 3234-3247.(2013)
4. Sala, G., Gobet, F.: Do the benefits of chess instruction transfer to academic and cognitive skills? A meta-analysis. In: Educational Reserch Review, vol. 18, pp 46-57. (2016)
5. Fürnkranz, J.: Knowledge Discovery in Chess Databases: A Research Proficial Intelligence. (1997)
6. Mathieu Acher, François Esnault.: Large-scale Analysis of Chess Games with Chess Engines: A Preliminary Report. [Technical Report] RT-0479, Inria Rennes Bretagne Atlantique. (2016)
7. León-Villagrá, P., Frank, J.: Categorization and Abstract Similarity in Chess. COGSCI . (2013)
8. E. Chavez and K. Figueroa. Faster proximity searching in metric data. Proceedings of MICAI 2004. LNCS 2972, Springer, (2004).
9. R. Baeza-Yates, W. Cunto, U. Manber, and S. Wu. Proximity matching using fixed-queries trees. In Proc. 5th Combinatorial Pattern Matching (CPM'94), LNCS 807, pages 198-212 (1994).
10. D. Novak, M. Batko, P. Zezula: Metric Index: An efficient and scalable solution for precise and approximate similarity search. Information Systems, vol. 36, issue 4, pp 721–733 (2011).
11. Santos, G.L., Rachelle, H.: ICONCHESS: An Interactive CONsultant for CHESS middlegames. ICLS '96 Proceedings of the 1996 i nternational conference on Learning sciences, pp. 456-461. (1996)
12. Hacohen-kerner, Y.: Computer Chess and Indexing. In: Encyclopedia of Computer Sciece and Technology, vol. 44, pp.59-80. (2001)
13. S. H. Cha. Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions. International Journal of Mathematical Models and Methods in Applied Sciences (2007).
14. H. W. Kuhn,: The Hungarian Method for the assignment problem. Naval Research Logistics Quarterly, 2: 83–97 (1955).
15. T. Skopal: On Fast Non-metric Similarity Search by Metric Access Methods. Advances in Database Technology - EDBT 2006. 10th International Conference on Extending Database Technology, Munich, Germany, pp 718-736 (2006)