

Algoritmo Genético de Chu-Beasley Aplicado a la Resolución del Viaje del Caballo de Ajedrez

Sergio Rubén Cossa*
Argentina - 2015

Palabras Clave: Algoritmo Genético, Chu-Beasley, Caballo de Ajedrez, Selección, Cruzamiento, Mutación, Cromosoma, Evolución, Inteligencia Artificial

Keywords: Genetic Algorithm, Chu-Beasley, Knight's Tour, Selection, Crossover, Mutation, Chromosome, Evolution, Artificial Intelligence

Introducción y Trabajos Previos

El Viaje del Caballo de Ajedrez es un problema que ha sido estudiado por siglos, tanto por jugadores de ajedrez como por matemáticos. Se plantea un recorrido del caballo, con su salto característico, de tal forma que recorra todas las casillas del tablero (64 casillas, 8x8) tocando cada una de ellas solo una vez.

El primer dato sobre este problema fue encontrado en un manuscrito árabe del año 840 (Murray, 1913) [1], donde se detallaban dos recorridos.

El primer análisis matemático fue presentado por Leonhard Euler en Berlín (1759) Otros matemáticos conocidos que lo trataron fueron Taylor, de Moivre y Lagrange.

Más cercanos a nuestra época, McKay [2] y Mordecki [3] usaron potentes mainframes buscando la mayor cantidad posible de recorridos sobre un tablero de 8x8. McKay calculó el total de recorridos cerrados (cuando el caballo completa el circuito, el próximo salto lo lleva a la casilla inicial) en 13.267.364.410.532, en tanto que Mordecki encontró 1.305×10^{35} recorridos abiertos.

En cuanto a los sistemas de búsqueda, van desde sistemas puramente heurísticos (Warnsdorff 1823), búsqueda en profundidad con retroceso (depth first search/backtracking), redes neuronales, algoritmos genéticos, algoritmos de colonias de hormigas, etc.

Para la resolución de este trabajo se construyó un algoritmo genético basado en el desarrollado por Chu-Beasley [4]

Algoritmos Genéticos

Los Algoritmos Genéticos (AGs) son métodos utilizados para resolver problemas de búsqueda y optimización. Se basan en la evolución de las poblaciones en la naturaleza, de acuerdo a los principios de selección natural y supervivencia del más fuerte, postulados por Charles Darwin (1859)

Al imitar esos procesos adaptativos, los AGs son capaces de generar soluciones para problemas de la vida real. Codificando de forma adecuada la población inicial del AG e implementando correctamente los métodos evolutivos, se puede llegar a soluciones óptimas.

Los principios básicos de los AGs fueron establecidos por Holland [5] en 1975, pero esa técnica recién se hizo popular a mediados de los '80, en especial por el trabajo de Goldberg [6] (1989)

En la naturaleza, los individuos compiten por los recursos de supervivencia, así como para lograr su reproducción. Los que tienen mejores condiciones para sobrevivir y atraer a sus parejas son los que más posibilidades obtendrán de generar descendientes. Por el contrario, los menos dotados, tendrán escasas posibilidades de reproducirse. Las características de los mejor adaptados son heredadas por sus hijos, quienes reciben una combinación de los mejores genes de sus padres y de ese modo evolucionan, logrando una mayor adaptación al entorno en que habitan.

Los AGs funcionan de forma similar al sistema natural. Cuentan con una población de individuos

donde cada uno es la solución potencial a un problema dado. Cada uno de esos individuos poseerá un valor de aptitud o adaptación relacionado con lo cerca o lejos que esté de lograr dicha solución.

Análogamente con la naturaleza, esos individuos podrán cruzarse y generar descendientes, los cuales recibirán lo mejor de sus ancestros y así, a medida que transcurren las generaciones, irán convergiendo hacia la solución óptima del problema.

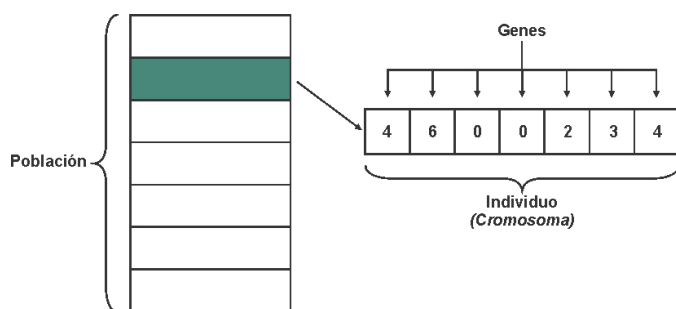
Un AG típico emplea nociones simplificadas de selección, cruzamiento, mutación y supervivencia del más apto para que, con un relativamente escaso conocimiento de los métodos para resolver un problema, de igual modo pueda llegar a una solución.

En los campos donde existen técnicas específicas para soluciones determinadas, estas siempre funcionarán mejor que un AG. La idea de la utilización de estos algoritmos es la búsqueda en campos sin una solución delimitada, o que si esa solución existe, se pueda mejorar combinándola con el uso del AG.

Los AGs son empleados en gran variedad de áreas, como ser: optimización numérica y combinatoria, aprendizaje de máquinas, programación automática, economía, sistemas inmunes, ecología, evolución y aprendizaje, sistemas sociales, etc.

Estructura y Funcionamiento de un Algoritmo Genético

Un AG Simple (como el creado por Holland y uno de los más utilizados) se compone de una población de individuos, también denominados cromosomas. Cada cromosoma representa una posible solución al problema propuesto y está compuesto por una serie de valores, los cuales pueden ser {0,1}, números enteros, reales, etc. Estos valores corresponden a cada uno de los genes que forman el cromosoma. Una posible representación gráfica es la siguiente:



El AG también cuenta con operadores genéticos, los cuales llevan adelante los procesos de selección, cruzamiento y mutación. Siendo esos tres

operadores los más comunes y básicos, algunos AGs especializados incorporan operadores de mejora de factibilidad, mejora de optimalidad, etc.

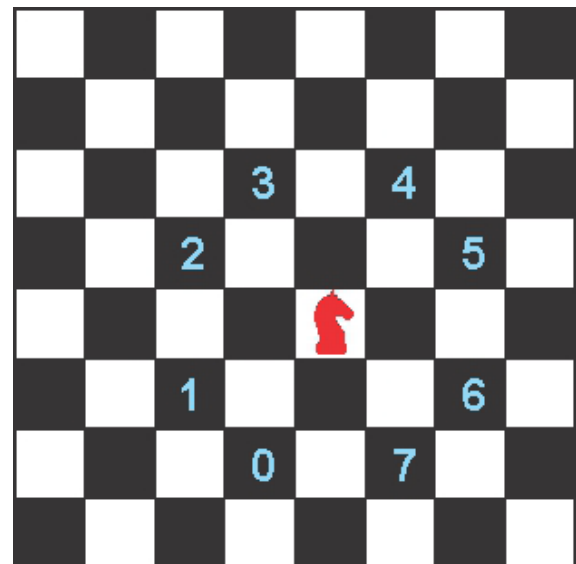
Algoritmo Genético de Chu-Beasley

Como se expresó más arriba, para este trabajo se utilizó el AG de Chu-Beasley. Las diferencias más importantes con los AGs simples son:

- *Diversidad.* No pueden existir cromosomas iguales en la población.
- En los AGs simples se reemplaza toda o casi toda la población en cada generación. En Chu-Beasley solo es reemplazado un individuo, siempre que el nuevo tenga las características deseadas de diversidad, factibilidad, etc.
- Incorpora un operador de mejora de optimalidad.

Adaptación del AG al Problema del Caballo de Ajedrez

Según la casilla en que esté ubicado, el caballo puede tener desde dos hasta ocho movidas legales, las cuales pueden ser codificadas de varias formas. En este trabajo se resolvió una codificación con enteros:



De ese modo, un recorrido parcial del caballo puede ser representado con una tira de enteros similar a esta:

$$4 - 6 - 0 - 0 - 2 - 4 \dots$$

Si tomamos un punto de partida como e4 (centro del tablero) y representamos ese recorrido en notación algebraica, tendríamos:

$$(e4) - f6 - h5 - g3 - f1 - d2 - e4$$

Nótese que el último movimiento nos lleva a una casilla ya visitada (e4) por lo que este recorrido es ilegal. Por esto, podríamos decir que este recorrido tiene una longitud válida de cinco saltos. Ese número de saltos es lo que denominamos **función de aptitud** del individuo. Para este problema, una función de aptitud de 63 saltos legales es una solución válida. Volviendo a la similitud con la naturaleza, un individuo o **cromosoma** es más o menos apto según si el número de su función de aptitud es mayor o menor. A mayor función de aptitud, más cerca se estará de solucionar el problema: completar el recorrido del caballo por todo el tablero de ajedrez.

Implementación del Algoritmo Genético

Se decidió crear una población inicial de 64 individuos o cromosomas (las pruebas posteriores corroboraron que es el número más apropiado) Cada uno de los genes de estos cromosomas fue completado con un número al azar entre 0 y 7. La única restricción inicial que se tomó fue que el salto desde una casilla x no llevara al caballo fuera del tablero.

Con la población inicial creada, se inicia un ciclo o generación, el cual se puede simplificar así:

- Selección de Progenitores
- Cruzamiento de Progenitores y generación de un hijo
- Mutación del hijo
- Mejora de adaptabilidad del hijo
- Incorporación (posible) del hijo a la población

Selección de Progenitores

Existen diversos métodos para seleccionar a los padres que generarán un descendiente de la población. Los más utilizados son:

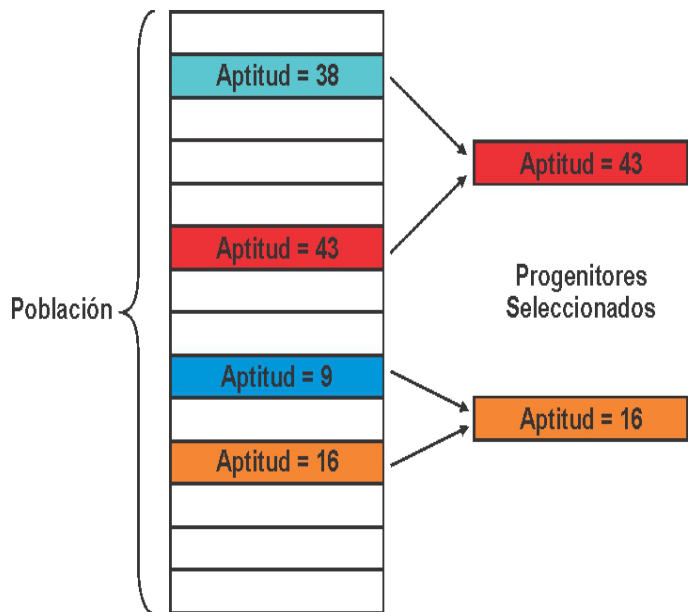
- Torneo
- Elitista
- Ruleta o Montecarlo
- Ranking

En este trabajo se experimentaron las selecciones por Torneo, Elitista y Montecarlo, siendo esta última la que mejores resultados produjo, como se verá más adelante.

Selección por Torneo

Se eligen dos pares de Progenitores Candidatos (cromosomas) al azar en la población. De cada par se elige el cromosoma que mejor función de aptitud

posee y de los dos elegidos se produce el cruzamiento.



Selección Elitista

Se toman siempre los dos mejores cromosomas de la población y se produce un cruzamiento entre ellos para generar el hijo.

Selección de Montecarlo o Ruleta

Con este método, la probabilidad que tiene un individuo de reproducirse es proporcional a la diferencia entre su aptitud y la de sus competidores. Esto puede representarse como un juego de ruleta donde cada individuo obtiene una sección o parte de la ruleta, pero los más aptos obtienen secciones mayores que las de los menos aptos. Luego, la ruleta se hace girar (se escoge un número al azar), y cada vez se elige al individuo que tenga la sección en la que se pare la ruleta.

Ejecutando dos veces la ruleta, se seleccionan los dos progenitores a cruzarse.

Cruzamiento de Progenitores

Con los dos cromosomas seleccionados se produce un cruzamiento de genes, de los cuales se generarán dos hijos. Solo el más apto de estos hijos podrá entrar a la población.

Así como existen diversos métodos de selección, también ocurre con el cruzamiento. En este trabajo se probaron los operadores de cruce por un punto y por dos puntos, resultando el cruce por un punto el que tuvo más éxito. Su implementación es la siguiente:

Una vez elegidos los dos progenitores, se cortan sus cromosomas por un punto seleccionado aleatoriamente. Con esto se generan dos segmentos

diferenciados en cada cromosoma: cabeza y cola. Luego se intercambian los genes de las colas entre cada individuo y de esa forma se crean dos hijos que heredan características de ambos progenitores. Finalmente solo el hijo con mejor función de aptitud tendrá posibilidades de ingresar a la población.

Progenitor 1	4	6	0	0	2	3	4
Progenitor 2	5	0	7	1	4	2	1
Hijo 1	4	6	0	0	4	2	1
Hijo 2	5	0	7	1	2	3	4

Mutación del Hijo

La mutación del hijo provoca que alguno de sus genes, generalmente uno solo, varíe su valor de forma aleatoria. Se imita de esta manera el comportamiento que se da en la naturaleza, pues cuando se genera la descendencia suele producirse algún tipo de error, por lo general sin mayor trascendencia, en el paso de la carga genética de padres a hijos. Normalmente la probabilidad de mutación es muy baja, menor al 1%. Sin embargo, las pruebas en este trabajo tuvieron mejores resultados con una mutación cercana al 85%.

Los operadores genéticos de mutación más utilizados son el de reemplazo aleatorio y el de intercambio de dos genes del cromosoma (swapping)

En el presente trabajo existe la posibilidad de que cualquiera de los dos operadores pueda ser aplicado en una mutación.

Cromosoma Hijo	4	6	0	0	2	3	4
Mutación Aleatoria	4	6	0	7	2	3	4
Cromosoma Hijo	4	6	0	0	2	3	4
Mutación por Swapping	4	2	0	0	6	3	4

Mejora de Adaptabilidad del Hijo

Este punto es más que importante en el AG. Las pruebas realizadas por varios autores (y también en el presente trabajo), con AGs simples, presentaron fracasos en la búsqueda de recorridos válidos para el caballo de ajedrez. A fin de mejorar el funcionamiento de los AGs se implementaron diversos métodos que especializan en parte el proceso. Esta idea de adaptabilidad fue explorada por Whitley [7] (1994)

Gordon y Slocum [8] (2004) introdujeron la técnica de reparación, la cual se explicará a continuación. Otra especialización fue la de Al-Gharaibeh, Qqwagneh y Al-zahawi [9] (2007), quienes mejoraron la adaptabilidad del descendiente mediante heurística.

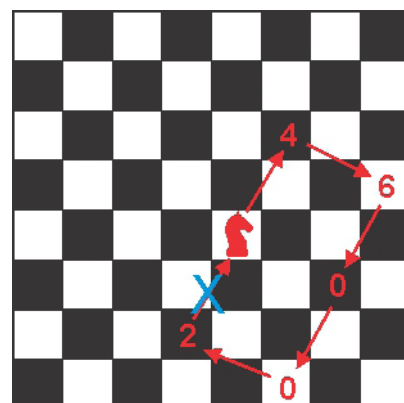
La idea de Gordon y Slocum fue la siguiente: para cada cromosoma de la población, en el punto donde un salto es inválido (es decir, lleva al caballo fuera del tablero o a una casilla ya visitada) se aplica una reparación. Se examina esa movida y se la intenta sustituir por otra movida válida y que permita seguir el viaje. Si no existe un salto válido, se termina la evaluación de ese cromosoma. En caso de que haya un salto válido, se modifica el gen inválido por ese salto y se pasa al gen que sigue a la derecha. Se prosigue así hasta que vuelva a aparecer una casilla sin saltos posibles o se llegue a la solución del recorrido del caballo. No se aplica ninguna heurística ni retroceso, como ocurre con otras técnicas.

En el ejemplo presentado más arriba, la evaluación del cromosoma finalizaba cuando el caballo saltaba nuevamente a la casilla e4. Esto es, la cadena de genes:

4 - 6 - 0 - 0 - 2 - 4...

(e4) - f6 - h5 - g3 - f1 - d2 - e4

donde el último valor (4) es inválido.



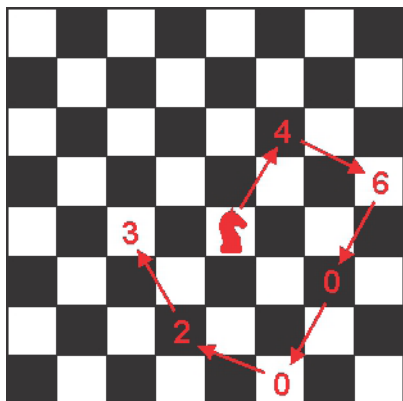
El proceso de reparación intenta encontrar un valor correcto para ese gen, digamos el valor 7, el cual nos sacaría fuera del tablero. Entonces se busca otro valor, por ejemplo 3, que nos llevaría a c4, la cual es

una casilla aún no visitada. Entonces la cadena de genes quedaría

4 – 6 – 0 – 0 – 2 – 3...

(e4) – f6 – h5 – g3 – f1 – d2 – c4

y se procedería del mismo modo con el gen siguiente a la derecha de la cadena.



La diferencia entre el presente trabajo y el de Gordon y Slocum es que ellos aplican la reparación sobre toda la población, en cambio en nuestro AG de Chu-Beasley trabajamos solo sobre el hijo recién creado.

Incorporación del Hijo a la Población

Una vez obtenido y mejorado el hijo, se calcula su **función de aptitud**, es decir, cuántos saltos válidos tiene su cadena de genes. Esa función es comparada con cada uno de los miembros de la población actual. Si la función de aptitud es mejor que la del peor miembro de la población, este individuo es eliminado y es reemplazado por el hijo actual. Siempre respetando la diversidad: que el hijo no sea igual a ninguno de los miembros presentes en la población.

Si en la población ya existiera un cromosoma con los mismos genes que el hijo actual, o si este hijo no fuera mejor que el peor de los individuos, entonces es descartado y se asume que no hubo descendencia en esta generación.

Resultados Obtenidos

Para el trabajo se utilizó el lenguaje C# junto al framework A-Forge.Net [10], un excelente producto gratuito y de código abierto, orientado a la Inteligencia Artificial y que permite acelerar los tiempos de desarrollo.

Se procuró efectuar un proceso de testeo similar a

las investigaciones anteriores, pero al ser Chu-Beasley un AG diferente, no se pueden utilizar los mismos parámetros. Finalmente, se creó una población inicial de 64 cromosomas y se efectuaron 1.000.000 de generaciones por cada casilla del tablero. El proceso se repitió cinco veces para cada casilla, a fin de obtener promedios comparables a las otras propuestas. Los resultados pueden observarse en la siguiente tabla:

Tipo de Selección	Generación Primer Viaje	Promedio Primera Generación con Viaje	Promedio de Viajes	Mayor Cantidad de Viajes	Menor Cantidad de Viajes
Montecarlo	1	36	222	529	33
Torneo	1	37	140	327	26
Elite	1	73	195	395	40

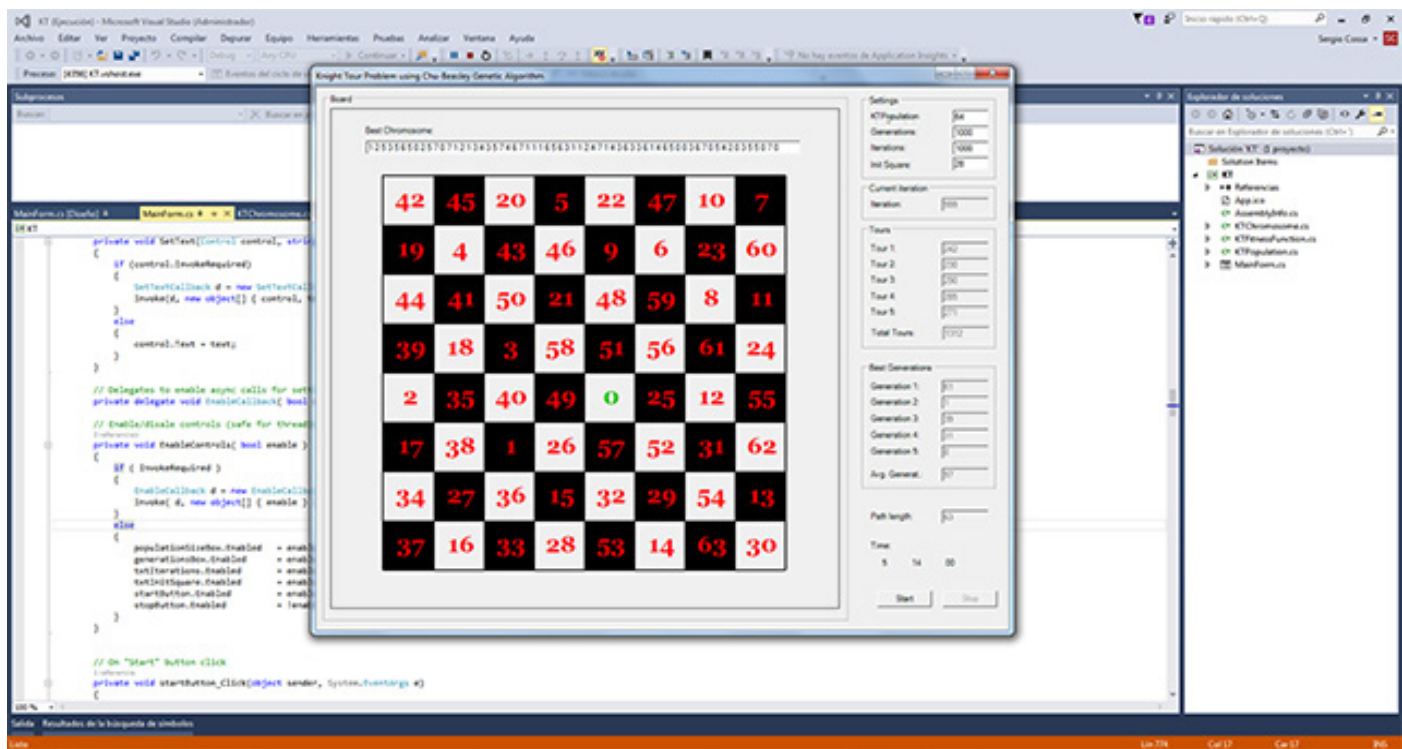
Los tres métodos de selección permitieron encontrar, al menos en una oportunidad, un viaje en la primera generación. Esto es gracias al mecanismo de reparación.

Es de notar que, si bien el AG de Gordon y Slocum encontró una mayor cantidad de recorridos en una casilla individual (642 contra 529), nuestro AG reconoció un promedio mayor en casi todas las casillas del tablero.

Por otra parte, no hubo pruebas que dieran resultado negativo, es decir, siempre el AG encontró recorridos válidos en cada testeo.

358	234	192	267	269	242	226	436
276	221	52	186	168	133	188	203
233	156	220	243	249	201	82	215
289	216	216	239	245	207	237	271
257	180	209	256	261	242	179	249
199	88	209	213	238	186	96	237
228	212	127	204	166	100	208	224
478	267	228	276	252	191	226	376

Promedios de Viajes en cinco tests.



Referencias:

- [1] Murray H.J.R. (1913) History of Chess
- [2] B. D. McKay, "Knight's tours of an 8x8 chessboard," Ph.D. dissertation, Department of Computer Science, Australian National University, 1997.
- [3] E. Mordecki, "On the number of knight's tours," in Prepublicaciones de Matematica de la Universidad de la Republica, Uruguay 2001/57, 2001.
- [4] BEASLEY, J.E. E CHU, P. C. A Genetic Algorithm for the Generalized Assignment Problem. Computers Operations Research, 24(1), pp 17-23, 1997.
- [5] Holland, J., Adaptation in Natural and Artificial Systems, 1st ed., Univ. of Michigan, 1975. 2nd ed. 1992 by MIT Press.
- [6] Goldberg, D., Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, 1989.
- [7] Whitley, D., Gordon, V.S., and Mathias, K., "Lamarckian Evolution, the Baldwin Effect and Function Optimization," Parallel Prob. Solving from Nature 3, Israel, pp 6-15, 1994
- [8] Gordon V.S. and Slocum T.J. (2004) The Knight's Tour – Evolutionary vs. Depth-First Search. In proceedings of the Congress of Evolutionary Computation 2004 (CEC'04), Portland, Oregon, pp. 1435-1440
- [9] J. Al-Gharaibeh, Z. Qawagneh, and H. Al-zahawi, "Genetic algorithms with heuristic – Knight's tour problem," in Proc. of International Conference on Genetic and Evolutionary Methods, 2007, pp. 177-81.
- [10] A-Forge.Net
<http://www.aforge.net/aforge/framework/>