Resumen Artículo Computación evolutiva

Camilo Andrés Rivera

19 de Junio de 2015

En todos los ámbitos en los cuales se utiliza el cómputo paralelo, el objetivo último es la mejora del rendimiento (ya sea de un problema usualmente resoluble en largos tiempos o de problemas antes impensables para su resolución). En últimas son problemas que exigen una capacidad de cómputo muy grande, ya sea por tamaño de datos o complejidad de cálculos. Un gran ejemplo de algoritmos en los que la paralelización ofrece una alternativa, y que además es una vertiente que ha venido ganando fama en los últimos años, es la expuesta en artículo; algoritmos genéticos, también conocida como computación evolutiva [Eiben and Smith, 2015].

La base de los algoritmos genéticos es en sí la evolución natural, puesto que para su realización se deben definir unos genotipos, la forma en que dichos genotipos se ligan con los correspondientes fenotipos (cómo se expresan) y finalmente una función de costo que será la métrica de desempeño del algoritmo. De manera que sólo los problemas correctamente planteados (o cuyas estructuras de datos estén correctamente distribuidas) pueden ser resueltos mediante la aplicación de esta técnica. Asimismo, el algoritmo es altamente escalable, ya que el proceso que lleva a cabo la *evolución* es ciego al problema que está resolviendo o desarrollando, tan solo se encarga de realizar una serie de generaciones y calcular la función de costo para cada individuo generado.

La idea de estos algoritmos es generar una familia inicial de posibles soluciones y de manera similar a la evolución, de dichos "padres" se obtiene la siguiente generación mediante la generación de mutaciones aleatorias y recombinaciones de dos o más padres. De esta manera se evalúa la función de costo en la nueva generación de soluciones. A pesar de tener un avance lento, este avance no se ve tan fuertemente afectado al incrementar el número de variables o datos, como sí lo están los algoritmos tradicionales, y además de esto no está ligado a los preceptos humanos existentes ante una solución; es decir, se pueden generar soluciones inimaginables. Otra ventaja de la computación evolutiva es que se generan una familia completa de soluciones, la cual además tiene baja posibilidad de quedarse atascada en un mínimo local entre generaciones. Esto último es de alto interés para los tipos de problemas en los cuales se tienen múltiples mínimos.

La tendencia de los últimos años es de no sólo utilizar los algoritmos genéticos para la evolución de códigos y soluciones computacionales (como lo son los software) sino a lo que se le llama la evolución de las cosas, como productos que están sujetos a una serie de requerimientos y para los cuales este camino ha generado nuevas soluciones. En el aspecto práctico, en donde no se requiere llegar a la solución óptima sino que una aproximación es útil, estos algoritmos son de gran utilidad, debido a que aportan una serie de soluciones cercanas al óptimo. Cabe la pena mencionar que en general los problemas de optimización pueden ser traducidos con facilidad a este algoritmo.

Se mencionan diversas áreas en las que se ha tenido éxito en su implementación, como por ejemplo la creación de nuevas antenas para la NASA, con la mejora de que genera una familia de soluciones probadas sin necesidad de fabricar tantas soluciones; para la generación de fármacos a partir de una base de datos de moléculas bioactivas, una gran cantidad de usos en la inteligencia artificial, como por ejemplo la creación de redes neronales; creación de controladores para robots y maquinarias; entre otros.

Es un área de gran trabajo y que aún es difícil generar una estandarización. Se tiene una amplia gama de trabajos por realizar, pero aún en esta etapa se están generando una gran variedad de usos para los algoritmos genéticos.

Referencias

[Eiben and Smith, 2015] Eiben, A. E. and Smith, J. (2015). From evolutionary computation to the evolution of things. *Nature*, 521(7553):476–482.