TẠP CHÍ ISSN: 1859-316X
KHOA HỌC CÔNG NGHỆ HÀNG HẢI
JOURNAL OF MARINE SCIENCE AND TECHNOLOGY

SONG SONG HÓA THUẬT TOÁN LAI GHÉP DAVIS' ORDER CROSSOVER TRÊN FPGA SỬ DỤNG TRUE DUAL PORT RAM - MỘT CÁCH TIẾP CẬN TRONG GIẢI QUYẾT BÀI TOÁN NGƯỜI DU LỊCH BẰNG GIẢI THUẬT DI TRUYỀN

PARALLEL DAVIS'S ORDER CROSSOVER USING TRUE DUAL PORT RAM OF FPGA - AN APPROACH TO SOLVE TRAVELLING SALESMAN PROBLEM BY GENETIC ALGORITHM

NGUYỄN TRUNG QUÂN, NGUYỄN TRỌNG ĐỨC*

Khoa Công nghệ thông tin, Trường Đại học Hàng hải Việt Nam

**Email liên hệ: trong-duc.nguyen@vimaru.edu.vn*

Tóm tắt

Bài toán Người du lịch (TSP - Travelling Salesman Problem) được xem là một trong những bài toán kinh điển của tối ưu hóa, đã và đang được ứng dụng rộng rãi trong nhiều lĩnh vực như lập kế hoạch, thiết kế vi mạch, phân tích gen,.. TSP với lời giải tổng quát thuộc lớp bài toán có độ phức tạp không phải đa thức (NP - đầy đủ), vì vậy việc tìm kiếm lời giải tối ưu cho bài toán là không khả thi. Đã có nhiều nghiên cứu nhằm nâng cao hiệu năng cho TSP trong phạm vi vài chục ngàn thành phố như sử dụng giải thuật tìm kiếm Tabu, mạng Nơron nhân tạo, giải thuật Di truyền (GA - Genetic Algorithm),.. Trong bài báo này, nhóm tác giả đề xuất giải pháp tăng cường mức độ song song hóa giải thuật GA nhằm cải thiện hiệu năng của giải thuật này khi giải quyết bài toán TSP bằng cách song song hóa thuật toán OX1 (Davis' Order Crossover) trên nền tảng FPGA (Field-Programmable Gate Array) với True Dual - Port RAM (T2P-RAM).

Từ khóa: *Bài toán Người du lịch, giải thuật di truyền, Davis' Order Crossover, FPGA.*

Abstract

Travelling Salesman Problem (TSP) is an optimization problem. It has several applications, such as scheduling, VLSI, logistics, supply chain optimization. TSP is a NP-Hard problem, so it is impossible to find an optimal solution in polynomial time. There is much research that improves TSP performance for thousands of cities, such as Tabu algorithm, neural network, genetic algorithm (GA). In this paper, we suggest a proposed solution to improve parallelization of GA in order to reduce processing time when applied to solve TSP by parallel OX11 step using True Dual Port RAM of the Field-Programmable Gate Array (FPGA).

Keywords: *Travelling Salesman Problem, Genetic Algorithm, Davis' Order Crossover, FPGA.*

1. Mở đầu

Bài toán Người du lịch (TSP - Travelling Salesman Problem) được xem là một trong những bài toán kinh điển của tối ưu hóa, đã và đang được ứng dụng rộng rãi trong nhiều lĩnh vực như lập kế hoạch, thiết kế vi mạch, phân tích gen,... [1]. Đã có nhiều nghiên cứu nhằm nâng cao hiệu năng cho TSP trong phạm vi vài chục ngàn thành phố như sử dụng giải thuật tìm kiếm Tabu [2], mạng Nơron nhân tạo [3], giải thuật Di truyền [4],... Tuy nhiên, giải pháp đưa ra hiện dừng lại ở các phương pháp tính toán và xử lý tuần tự, chưa khai thác được thế mạnh của các giải thuật tính toán song song cũng như sự phát triển của kĩ thuật phần cứng.

Tính toán song song hiệu năng cao được xem là một xu thế phát triển cho các hệ thống hiện nay. Lập lịch và đồng bộ giữa các tác vụ là một trong các thách thức để có thể song song hóa quá trình xử lý [5]. Một số mô hình đã được đề xuất khi thực hiện song song hóa các phần mềm trên nền tảng của CPU hoặc GPU [6]. Tuy nhiên, việc song song hóa này tồn tại những hạn chế nhất định về mặt thời gian khi thực hiện trao đổi dữ liệu và đồng bộ giữa các tiến trình [7]. Khi đó, giải pháp đề ra là triển khai song song hóa trực tiếp trên các hệ thống phần cứng thông qua việc tối ưu thiết kế cho các cơ chế giao tiếp của các đơn vị xử lý. Giải pháp thực hiện song song hóa giải thuật GA trên nền tảng FPGA cho bài toán TSP được đề xuất. Trong bài báo này, nhóm tác giả đề xuất giải pháp tăng cường mức độ song song hóa giải thuật GA nhằm cải thiện hiệu năng của giải thuật này khi giải quyết bài toán TSP bằng cách song song hóa thuật toán OX1 trên FPGA với True Dual Port RAM (T2P-RAM).

Nội dung bài báo bao gồm: Mục 1 - Mở đầu; mục 2 - Mô hình kiến trúc hệ thống; Mục 3 - Song song hóa thuật toán OX1 sử dụng T2P- RAM và cuối cùng là Kết luận và hướng nghiên cứu tiếp theo của nhóm.

2. Mô hình kiến trúc hệ thống

Mô hình kiến trúc hệ thống được chỉ ra trong Hình 1.

Khối Core: Đảm nhiệm nhiệm vụ chính để tương tác giữa các luồng dữ liệu, đồng thời chứa quần thể khởi tạo ban đầu.

Khối Logger: Nhận dữ liệu về cá thể tốt nhất sau mỗi thế hệ, lưu trữ vào bộ nhớ.

Khối RandSelection: Tạo ra 2 giá trị ngẫu nhiên là vị trí trong quần thể nhằm lựa chọn hai cá thể ban đầu.

Khối Tournament: Kết hợp dữ liệu từ hai khối RandSelection để lựa chọn ra cá thể đưa vào quá trình lai ghép theo giải thuật 2-way tournament.

Khối Replacement: Xác định vị trí các cá thể bị thay thế bởi các cá thể mới.

Khối InvariantCtrl: Sinh hai giá trị ngẫu nhiên theo thứ tự nhằm chọn ra hai điểm trên chuỗi gen phục vụ cho quá trình trao đổi chéo và đột biến.

Cụm Computing: Bao gồm nhiều khối xử lý nhỏ hoạt động theo cơ chế đường ống nhằm hỗ trợ việc tạo ra cá thể mới thông qua việc trao đổi chéo và đột biến.

Trong đó:

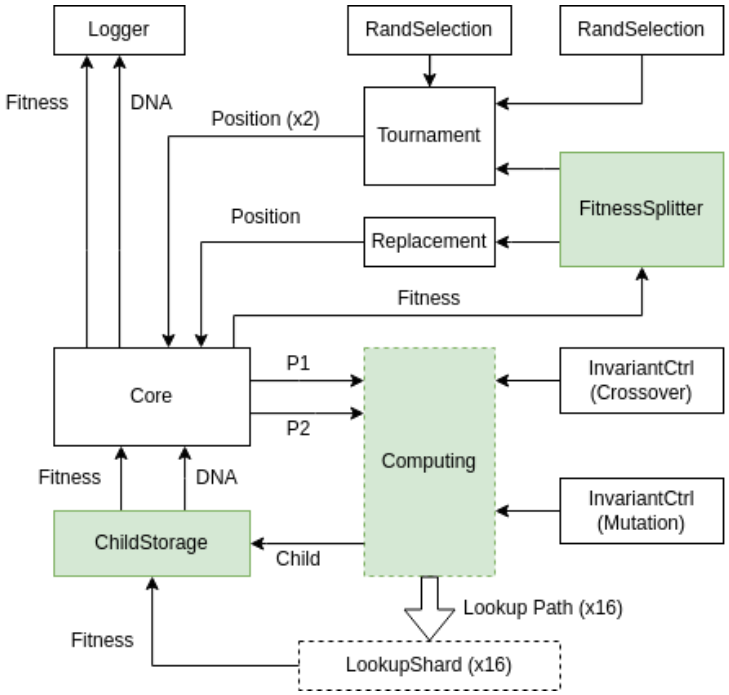
Khối OX1: Thực hiện trao đổi chéo dựa theo giải thuật OX1.

Khối Invert: Thực hiện đột biến chuỗi gen dựa theo giải thuật đảo ngược.

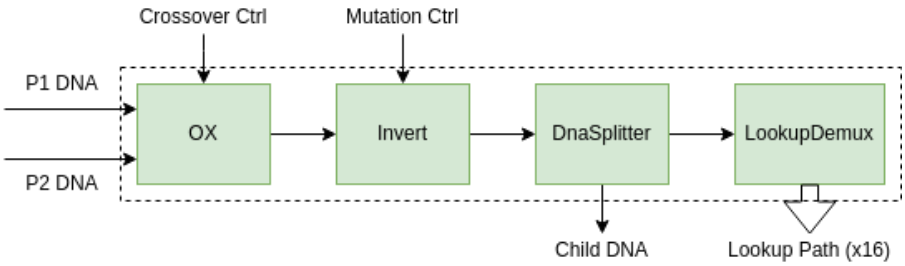
Khối DnaSplitter: Nhân bản luồng dữ liệu Nhiễm sắc thể (DNA) thành hai luồng giống nhau với một luồng đưa ra ngoài và luồng còn lại đi tới khối LookupDemux.

Khối LookupDemux: Phân tách chuỗi DNA mã hóa chu trình thành các đoạn đường cần di chuyển và phân loại chúng về các luồng tìm kiếm chi phí cho đoạn đó.

Mỗi khối trong cụm được triển khai thành một đơn vị xử lý độc lập trên FPGA, đồng thời chúng cũng



Hình 1. Mô hình kiến trúc hệ thống



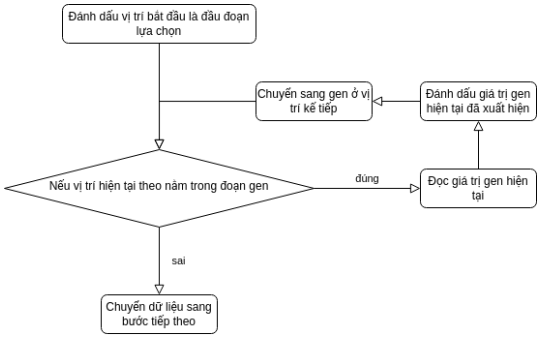
Hình 2. Sơ đồ khối cụm Computing

TẠP CHÍ ISSN: 1859-316X
KHOA HỌC CÔNG NGHỆ HÀNG HẢI
JOURNAL OF MARINE SCIENCE AND TECHNOLOGY

được lập lịch để hoạt động đồng bộ. Thêm vào đó, các khối được kết nối với nhau thông qua một hoặc nhiều kết nối AXI Stream nhằm chuyển dữ liệu theo một chiều từ khối này sang khối khác. Từ đó, cho phép việc xử lý của các khối này được song song ở mức tác vụ theo cơ chế đường ống. Nhờ vậy, việc cân bằng tốc độ xử lý của các tác vụ trong đường ống sẽ đảm bảo hiệu quả khi không có tác vụ nào phải chờ đợi được xử lý.

3. Song song hóa thuật toán OX1 sử dụng True Dual port RAM

Lưu đồ thuật toán OX1 tuần tự như chỉ ra trong Hình 3.



Hình 3. Lưu đồ thuật toán OX1 tuần tự

Với các bước:

Bước 1: Đánh dấu điểm bắt đầu xử lý là điểm đầu trong đoạn gen được chọn;

Bước 2: Kiểm tra nếu điểm cần xử lý nằm trong đoạn gen được chọn chuyển sang bước 3, nếu sai thì chuyển sang bước 6;

Bước 3: Đọc giá trị gen hiện tại;

Bước 4: Đánh dấu giá trị gen hiện tại đã xuất hiện;

Bước 5: Chuyển sang gen ở vị trí kế tiếp và quay về bước 2;

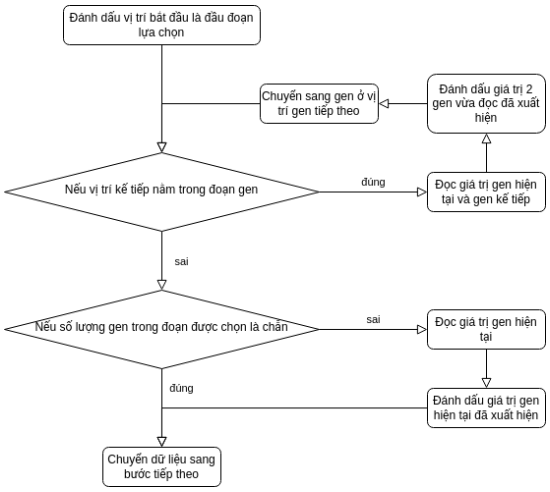
Bước 6: Chuyển dữ liệu sang phần xử lý kế tiếp.

Bộ nhớ hiện đang sử dụng trong khối OX1 có thể cấu hình ở chế độ làm việc T2P-RAM. Điều này cho phép việc đọc ghi có thể diễn ra đồng thời trên cả hai cổng của bộ nhớ RAM. Để tận dụng được lợi thế này, nhóm thực hiện song song hóa giải thuật OX1 trên nền tảng FPGA, lưu đồ thuật toán song song hóa OX1 được chỉ ra trong Hình 4.

Với các bước:

Bước 1: Đánh dấu điểm bắt đầu xử lý là điểm đầu trong đoạn gen được chọn;

Bước 2: Kiểm tra nếu điểm kế tiếp nằm trong đoạn gen được chọn chuyển sang bước 3, nếu sai thì chuyển sang bước 6;



Hình 4. Lưu đồ thuật toán OX1 song song

Bước 3: Đọc giá trị gen hiện tại và gen kế tiếp;

Bước 4: Đánh dấu giá trị 2 gen vừa đọc đã xuất hiện;

Bước 5: Chuyển sang gen ở vị trí kế tiếp và quay về bước 2;

Bước 6: Kiểm tra số lượng gen trong đoạn được chọn là chẵn thì chuyển sang bước 9, nếu sai thì thực hiện bước 7;

Bước 7: Đọc giá trị gen hiện tại;

Bước 8: Đánh dấu giá trị gen vừa đọc đã xuất hiện;

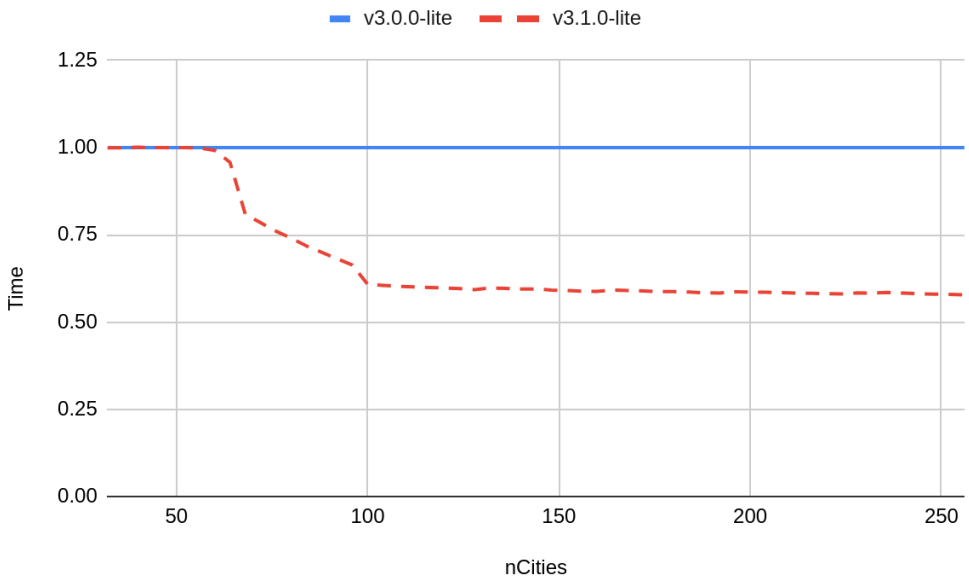
Bước 9: Chuyển dữ liệu sang phần xử lý tiếp theo.

Ở các bước 3 và bước 4 sẽ tận dụng bộ nhớ T2P-RAM để thực hiện được thao tác đọc và ghi được 2 ô nhớ một cách đồng thời. Điều này sẽ giúp giảm số chu kỳ cần thực hiện đi khoảng một nửa. Tuy nhiên, cũng cần bổ sung thêm bước xử lý bổ sung khi số lượng gen trong đoạn cần chọn là lẻ.

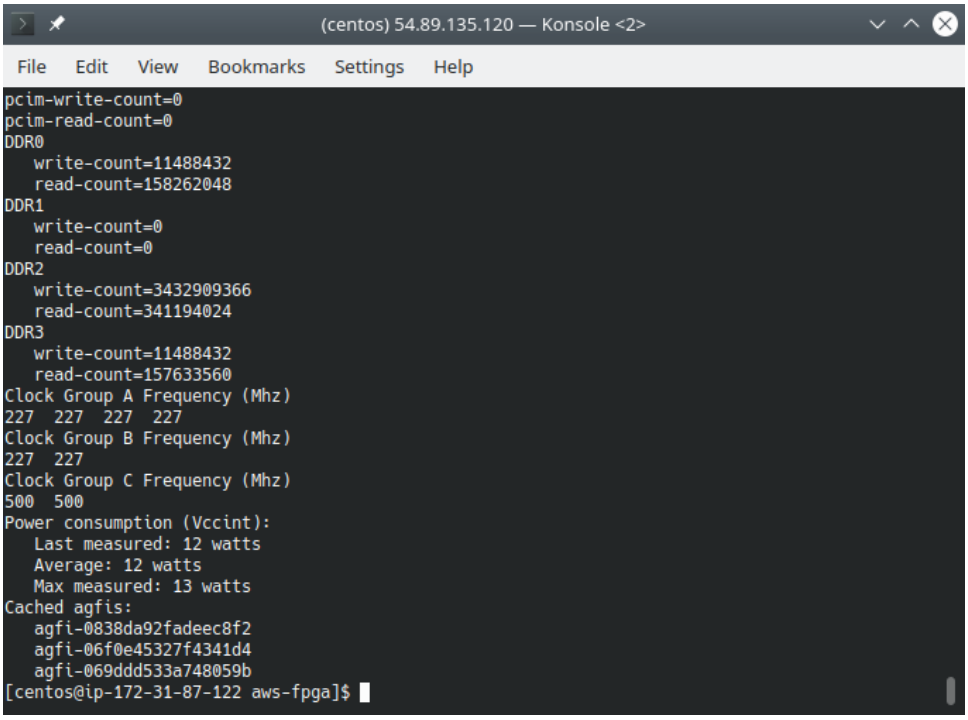
Các phần còn lại trong các khối xử lý của OX1 được thiết kế lại theo phương pháp tương tự để có thể tận dụng được lợi thế của T2P-RAM trên FPGA cho phép đọc ghi đồng thời trên cả hai cổng của bộ nhớ RAM.

4. Kết luận

Như đã trình bày trong mục trước, TSP tổng quát là bài toán thuộc lớp NP-đầy đủ. Để tối ưu hóa bài toán TSP tổng quát cho tới thời điểm hiện tại là không khả thi. Để kiểm chứng cho đề xuất của mình, nhóm tiến hành thử nghiệm cài đặt giải thuật OX1 song song đã đề xuất (trong phạm vi 250 thành phố), so sánh tốc độ tính toán và xử lý của giải thuật này với giải thuật OX1 tuần tự. Dữ liệu cho mẫu thử được sinh ngẫu nhiên cho các lần thử với 10000 nhiễm sắc thể trong quần thể để đánh giá hiệu năng của đơn vị xử lý cho



Hình 5. So sánh thời gian tính toán, xử lí bài toán TSP bởi OX1 song song và tuần tự



Hình 6. Thông số hệ thống khi chạy thử nghiệm

giả thuật OX1 được thiết kế trên phần cứng FPGA. Hình 5 chỉ ra sự so sánh về thời gian thực hiện giải thuật OX1 tuần tự (phiên bản v3.0.0-lite) và OX1 khi được song song hóa.

Biểu đồ trên Hình 5 cho thấy, thời gian xử lý trên OX1 song song chỉ chiếm khoảng 60% thời gian xử lý so với giải thuật tuần tự. Điều này khẳng định giải thuật OX1 sau khi tận dụng lợi thế của T2P-RAM đã giảm thời gian xử lý được gần 2 lần. Đặc biệt, khi số lượng các thành phố tăng lên, hiệu năng của giải thuật được cải thiện rõ rệt.

Bên cạnh đó, các thông số hệ thống trong quá trình chạy thử nghiệm của phiên bản v3.1.0-lite (Hình 6) cũng cho thấy tần số làm việc của đơn vị xử lý được thiết kế trên FPGA (B0) và tốc độ truyền dữ liệu giữa FPGA và máy tính (A0) hoàn toàn đồng bộ (227 MHz).

TẠP CHÍ ISSN: 1859-316X
KHOA HỌC CÔNG NGHỆ HÀNG HẢI
JOURNAL OF MARINE SCIENCE AND TECHNOLOGY

Đồng thời, mức độ tiêu thụ năng lượng trung bình của thiết kế trong quá trình thử nghiệm (~13W) thấp hơn nhiều so với các bộ xử lý thông thường thiết kế cho máy tính cá nhân (~45W). Điều này khẳng định sự kết hợp giữa FPGA với các máy tính thông thường làm gia tăng hiệu năng cho hệ thống, khi đó FPGA đóng vai trò như một bộ đồng xử lý cho các hệ thống này. Thêm vào đó, việc xử lý các tác vụ hay các bài toán con (bài toán lai ghép trong GA, nhận dạng ảnh,...) có thể được xử lý một cách độc lập trên các hệ FPGA chuyên dụng là hoàn toàn khả thi.

TÀI LIỆU THAM KHẢO

[1] Lê Anh Vinh, *Lý thuyết đồ thị*, NXB ĐHQG Hà Nội, 2020.

[2] C.N. Fiechter, *A Parallel Tabu Search Algorithm for Large Scale Traveling Salesman Problems*, Working Paper 90/1 Department of Mathematics, École Polytechnique Fédérale de Lausanne, Switzerland, 1990.

[3] V. Potvin, *The Traveling Salesman Problem: A Neural Network Perspective*, INFORMS Journal on Computing Vol.5, pp 328-348, 1993.

[4] J.V. Potvin, *Genetic Algorithms for the Traveling Salesman Problem*, Annals of Operations Research, Vol.63, pp.339-370, 1996.

[5] Nourah Al-Angari, Abdullatif ALAbdullatif, *Multiprocessor Scheduling Using Parallel Genetic Algorithm*, International Journal of Computer Science Issues, Vol. 9, Issue 4, No 3, pp.260-264, July 2012.

[6] ErickCantú-PazaDavid E.Goldbergb, *Efficient parallel genetic algorithms: theory and practiceComputer*, Methods in Applied Mechanics and Engineering, Vol.186, Issues 2-4, June 2000.

[7] John Runwei Cheng and Mitsuo Gen, *Parallel Genetic Algorithms with GPU Computing*, Intechopen, 2020.

Ngày nhận bài:	22/12/2021
Ngày nhận bản sửa:	29/12/2021
Ngày duyệt đăng:	02/01/2022