

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA



Đồ án Tổng hợp - Trí tuệ Nhân tạo (CO3101)
GVHD: Trịnh Văn Giang

ĐỀ TÀI DỰ ÁN P1

BDD Visualizer - Minh họa trực quan cấu trúc dữ liệu
Binary Decision Diagrams

Thành viên: Lương Minh Thuận - 2313348

Lê Doãn Thọ - 2313327

Lưu Chấn Kiến - 2211740

Mục lục

1. Tổng quan.....	3
1.1 Giới thiệu về Binary Decision Diagram (BDD).....	3
1.2 Hệ thống hiển thị đồ thị BDD	5
2. Mục tiêu	7
3. Kỳ vọng.....	7
4. Yêu cầu chức năng/phi chức năng.....	8
4.1 Yêu cầu chức năng:	8
4.2 Yêu cầu phi chức năng:	9
5. Các công cụ/công nghệ.....	10
6. Kết quả đầu ra	10

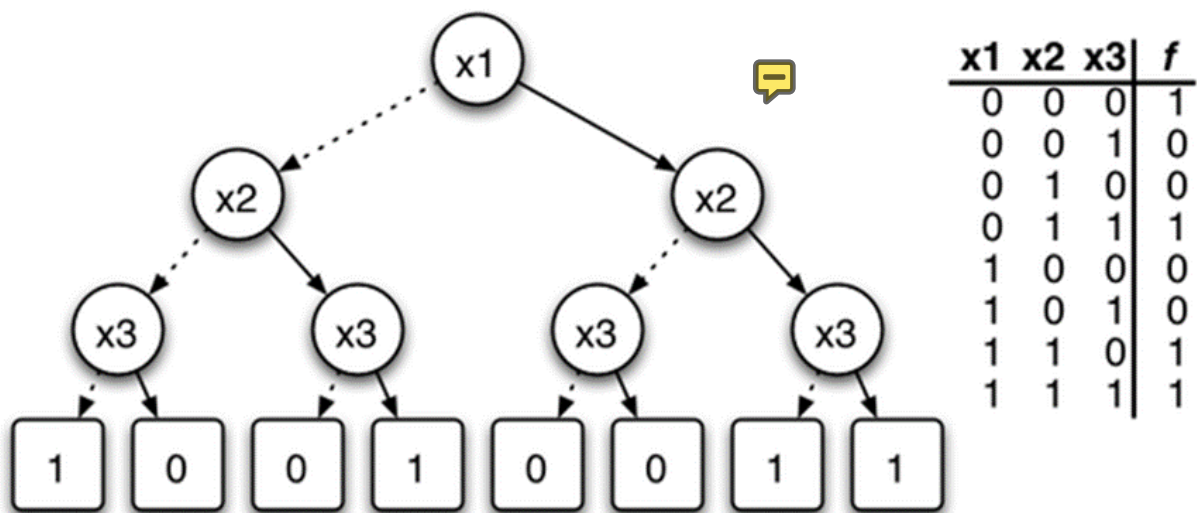
1. Tổng quan

1.1 Giới thiệu về Binary Decision Diagram (BDD)

Binary Decision Diagram (BDD) là một cấu trúc dữ liệu **đồ họa (graph)** có hướng, không có chu trình (directed acyclic graph – DAG), dùng để biểu diễn các hàm **Boolean**

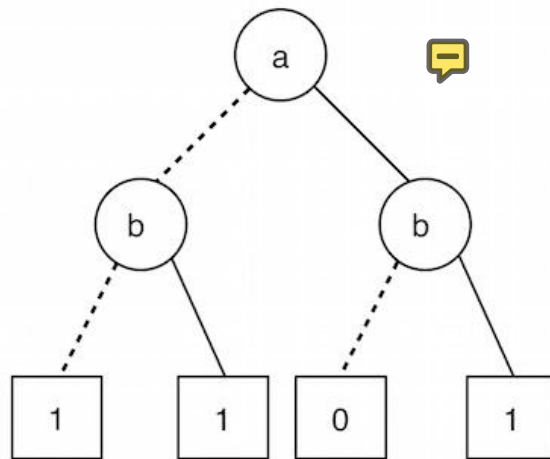
Một BDD có hai loại nút chính:

- Nút quyết định (decision nodes): mỗi nút được gán một biến Boolean, và từ đó có hai cạnh (edges) đi ra – một cho trường hợp biến = 0 (low child), một cho trường hợp biến = 1 (high child).
- Nút lá (terminal nodes): thường là các giá trị hằng như 0 và 1, biểu diễn kết quả của hàm Boolean khi một đường đi từ nút gốc (root) tới nút lá được chọn.

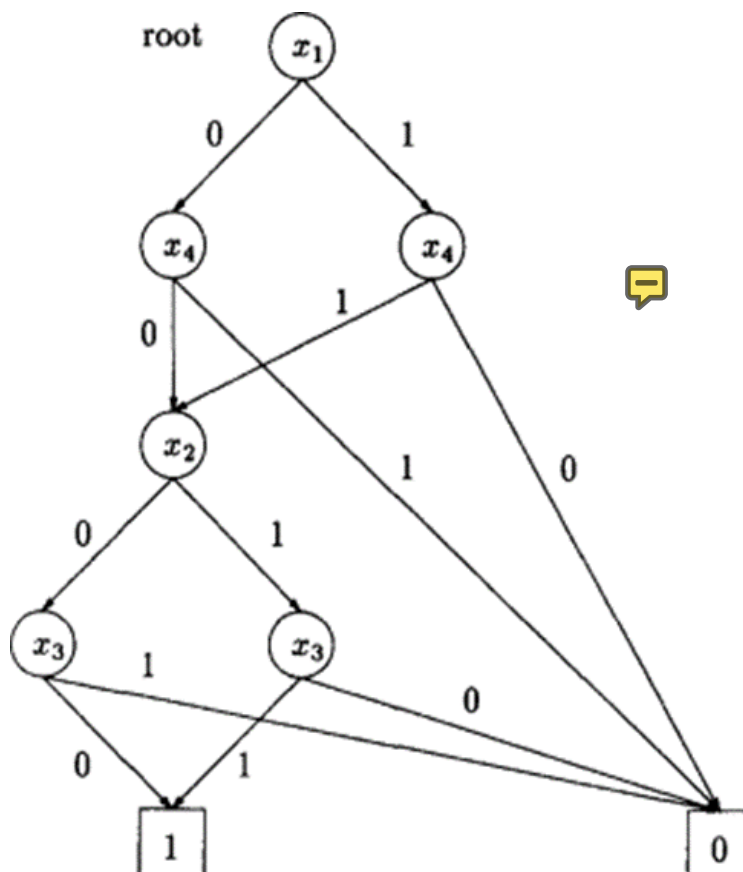


Một BDD được gọi là:

- Ordered BDD (OBDD): nếu biến (variable) xuất hiện theo cùng một thứ tự dọc theo mọi đường đi từ nút **root** đến nút lá.



- Reduced Ordered BDD (ROBDD): là OBDD đã được rút gọn theo các quy tắc nhất định để loại bỏ dư thừa và đạt được dạng tối giản. Một đặc điểm quan trọng của ROBDD là tính chuẩn hóa: với cùng một hàm Boolean và cùng một thứ tự biến, thì ROBDD sinh ra là duy nhất (unique). Nói cách khác, nếu hai ROBDD đại diện cho cùng hàm và tuân theo cùng ordering biến, chúng sẽ giống nhau về cấu trúc.



Ưu điểm, hạn chế, và ứng dụng của BDD

• **Ưu điểm:**

- BDD cho phép biểu diễn hàm Boolean một cách hiệu quả hơn so với biểu thức thông thường hay **truth table**, đặc biệt khi có nhiều biến nhưng cấu trúc hàm có tính lặp lại hoặc chuẩn hóa tốt.

- Các phép toán Boolean như AND, OR, NOT, ITE (if-then-else) có thể được thực hiện trực tiếp trên BDD thông qua các thuật toán đệ quy hoặc dựa vào chia nhỏ hàm theo Shannon expansion.

• Hạn chế:

- Kích thước của BDD phụ thuộc rất mạnh vào thứ tự biến (variable ordering). Nếu **ordering** chọn không tốt, số nút có thể nở theo hàm mũ – làm chi phí lưu trữ và tính toán tăng cao đáng kể.

- Một số hàm Boolean đơn giản vẫn có BDD lớn nếu không tìm được **ordering** tối ưu.

• Ứng dụng:

- Kiểm tra tính tương đương (equivalence checking) giữa hai hàm hoặc mạch logic.

- Mô phỏng symbolic (symbolic simulation) trong **verification** mạch số,

- Giải quyết bài toán SAT, đếm số nghiệm, xử lý biểu thức với biến lượng tử.

- Trong thiết kế mạch (logic synthesis), phân tích fault tree, lập kế hoạch trong AI, ...

1.2 Hệ thống hiển thị đồ thị BDD

Binary Decision Diagrams

This is [obst](#), a visualisation of algorithms related to [Binary Decision Diagrams](#), written by Philipp Czermer in 2018.

Read the help for more information, or get started right away by pressing "Create and add".

Hint: You can hover over nodes using your cursor, showing additional details.

Input type: ☐ List of numbers ☒ Boolean formula

$f = x_2 \wedge x_4$
 $(x_1 \& f) \leftrightarrow \neg x_3$

Base: 10

Variable order: auto

Adds the BDD to the graph.

Create and add

Operation: ☒ Union ☐ Intersection ☐ Complement

First node: a

Second node: f

Applies the operation.

Calculate union

Reset the application, delete all nodes.

Remove all

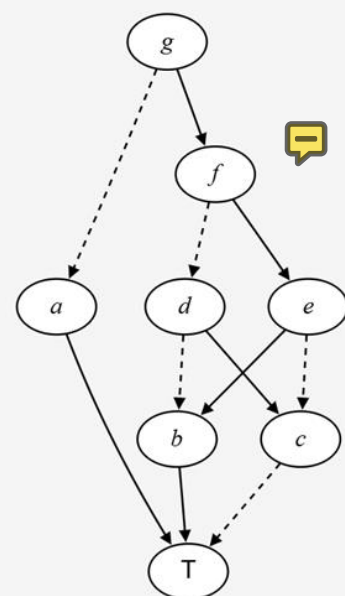
Display usage instructions.

Show help

Step-by-step (Move using arrow keys)

< 39/39 >

Done.



(source: <https://nicze.de/philipp/bdds/>)

Một trong những thách thức lớn khi làm việc với Binary Decision Diagram (BDD) là trực quan hóa cấu trúc của nó. Do BDD là một đồ thị có hướng vô chu trình (DAG), với nhiều nút và cạnh biểu diễn mối quan hệ logic, việc hiển thị giúp người dùng dễ dàng quan sát, phân tích và kiểm chứng hàm Boolean mà BDD đang biểu diễn. Hệ thống hiển thị đồ thị BDD thường cung cấp giao diện trực quan để hiển thị các nút (gốc, nút quyết định, nút lá) và các cạnh (0-edge, 1-edge) theo cách rõ ràng và có tính phân cấp.

Một hệ thống hiển thị đồ thị BDD thường bao gồm:

- **Nút gốc (root node):** thường đặt ở phía trên cùng của đồ thị.
- **Nút quyết định (decision nodes):** hiển thị tên biến Boolean gắn với nút, kèm theo hai cạnh đi ra tương ứng với giá trị 0 và 1.
- **Nút lá (terminal nodes):** biểu diễn giá trị 0 hoặc 1, thường hiển thị ở phía dưới cùng và được vẽ dưới dạng ô vuông/đỉnh đặc biệt.
- **Cạnh (edges):** hệ thống có thể dùng màu sắc hoặc kiểu nét khác nhau để phân biệt cạnh ứng với giá trị 0 (ví dụ nét đứt, màu xanh) và giá trị 1 (ví dụ nét liền, màu đỏ).

Ngoài ra, nhiều hệ thống còn hỗ trợ thu gọn (collapse) các nút lặp lại hoặc hợp nhất các nút đồng cấu để người dùng dễ quan sát hơn.

Các hệ thống hiển thị BDD hiện đại thường tích hợp nhiều tính năng:

- Tương tác động (interactive exploration): cho phép người dùng phóng to, thu nhỏ, kéo thả đồ thị để quan sát các phần chi tiết.
- Highlight đường đi: hiển thị đường đi từ nút gốc đến một nút lá tương ứng với một mẫu gán biến cụ thể.
- Tích hợp thuật toán: hiển thị trực quan quá trình rút gọn từ BDD sang ROBDD, giúp người học hiểu rõ hơn về các bước hợp nhất nút và loại bỏ dư thừa.
- Xuất dữ liệu: hỗ trợ xuất đồ thị sang các định dạng phổ biến như PDF, PNG hoặc JSON để phục vụ cho báo cáo hoặc xử lý tiếp theo.

Nhờ khả năng trực quan, hệ thống hiển thị BDD được dùng nhiều trong:

- Giảng dạy và học tập: hỗ trợ sinh viên và nhà nghiên cứu nắm bắt cách BDD biểu diễn hàm **Boolean**.


- Phân tích và xác minh mạch số: giúp kỹ sư logic kiểm chứng nhanh tính tương đương giữa các biểu thức.
- Công cụ hỗ trợ nghiên cứu: trong các lĩnh vực như AI, SAT solving, symbolic model checking, nơi trực quan hóa cấu trúc logic là rất quan trọng.

2. Mục tiêu

Mục tiêu chính của ứng dụng là cung cấp một công cụ trực tuyến thân thiện, nơi người dùng có thể nhập trực tiếp một công thức Boolean bất kỳ (theo dạng chuẩn, ví dụ: $(x \wedge y) \vee \neg z$). Hệ thống cần có khả năng phân tích cú pháp (parsing) công thức, chuyển đổi nó thành dạng chuẩn (như cây cú pháp), sau đó sinh ra đồ thị Binary Decision Diagram (BDD) tương ứng. Qua đó, ứng dụng giúp người dùng hiểu rõ cách BDD biểu diễn các hàm logic một cách trực quan và có hệ thống.

Một yêu cầu quan trọng là hiển thị đồ thị BDD một cách trực quan. Hệ thống cần biểu diễn rõ ràng các nút gốc, nút quyết định, nút lá, cùng với các cạnh biểu diễn giá trị 0 và 1. Ngoài ra, ứng dụng phải hỗ trợ tương tác động, như phóng to, thu nhỏ, kéo thả để quan sát toàn bộ đồ thị, hoặc highlight đường đi tương ứng với một mẫu gán biến cụ thể. Điều này đảm bảo trải nghiệm người dùng dễ dàng, ngay cả khi BDD có nhiều nút.

Ứng dụng không chỉ dừng lại ở việc sinh ra và hiển thị BDD, mà còn hướng tới việc hỗ trợ giảng dạy và nghiên cứu. Cụ thể, hệ thống có thể cho phép lựa chọn hiển thị BDD dạng chuẩn (BDD thô), hoặc dạng ROBDD (Reduced Ordered BDD) sau khi áp dụng các quy tắc rút gọn. Tính năng này giúp người học quan sát quá trình biến đổi, từ đó hiểu rõ hơn về khái niệm chuẩn hóa và tối ưu BDD.

Một mục tiêu khác là khả năng xuất và chia sẻ kết quả. Hệ thống nên hỗ trợ xuất đồ thị sang các định dạng phổ biến như hình ảnh (PNG, SVG) hoặc  liệu (JSON, DOT/Graphviz) để phục vụ cho báo cáo, tài liệu học tập hoặc nghiên cứu. Ngoài ra, ứng dụng có thể phát triển thêm API hoặc module tích hợp, giúp sinh viên, nhà nghiên cứu, hoặc kỹ sư dễ dàng nhúng công cụ này vào hệ thống lớn hơn.

3. Kỳ vọng

a. Học thuật và lý thuyết:

Xây dựng một công cụ hỗ trợ trực quan hóa giúp sinh viên, giảng viên và nhà nghiên cứu có thể dễ dàng tiếp cận và hiểu rõ hơn về cấu trúc dữ liệu Binary Decision Diagram (BDD).

Góp phần làm rõ vai trò của BDD trong việc biểu diễn và tối ưu các hàm Boolean, từ đó củng cố kiến thức nền tảng về logic, cấu trúc dữ liệu và các ứng dụng trong thiết kế mạch số, kiểm chứng mô hình (model checking) và lĩnh vực trí tuệ nhân tạo.

b. Ứng dụng và công nghệ:

Phát triển một ứng dụng web đơn giản, thân thiện, có khả năng tự động hóa quá trình chuyển đổi từ công thức Boolean sang đồ thị BDD, thay thế cho việc vẽ tay thủ công vốn tốn thời gian và dễ sai sót.

Cung cấp giao diện trực quan, hỗ trợ các thao tác cơ bản như phóng to/thu nhỏ, kéo thả, và xuất đồ thị dưới dạng hình ảnh, từ đó nâng cao trải nghiệm người dùng.

c. Đào tạo và thực tiễn:

Giúp nhóm sinh viên rèn luyện khả năng kết hợp giữa lý thuyết thuật toán và thực hành lập trình, đồng thời phát triển kỹ năng làm việc nhóm, quản lý tiến độ và áp dụng công cụ hỗ trợ phát triển phần mềm.

Sản phẩm có thể được sử dụng như một công cụ giảng dạy minh họa trong các môn học liên quan đến Toán rời rạc, Logic số, Cấu trúc dữ liệu và Thiết kế mạch số.

4. Yêu cầu chức năng/phi chức năng

4.1 Yêu cầu chức năng:

a. Boolean Formula Input

- Hệ thống cho phép người dùng nhập công thức Boolean bằng ký hiệu chuẩn hoặc ký hiệu toán tử (ví dụ: $x1 \text{ AND } (x2 \text{ OR NOT } x3)$ hoặc $x1 \& (x2 \wedge \neg x3)$).
- Có hỗ trợ các phép toán cơ bản: AND, OR, NOT, XOR, IMPLIES, EQUIVALENT.
- Cho phép nhập bằng giao diện text box và kiểm tra cú pháp.

b. BDD Generation

- Hệ thống chuyển công thức Boolean sang biểu diễn BDD.
- Tự động thực hiện quá trình rút gọn (Reduced Ordered BDD – ROBDD) để tối giản số node.
- Có khả năng xử lý công thức với số lượng biến vừa phải ($\leq n$ biến) ($n=10$ hoặc 15).

c. BDD Visualization

- Hiển thị BDD dưới dạng đồ thị nút – cạnh.
- Node biểu diễn biến, cạnh trái/phải tương ứng với giá trị 0/1.

- Có màu sắc, label rõ ràng (ví dụ: cạnh “0” màu đỏ, cạnh “1” màu xanh).

d. Graph Interaction

- Hỗ trợ phóng to, thu nhỏ, kéo thả để di chuyển sơ đồ.
- Cho phép highlight đường đi theo một assignment (ví dụ: gán $x1=1$, $x2=0$, ... sẽ hiển thị đường đi đến kết quả).

e. Export Options

- Cho phép xuất sơ đồ BDD ra định dạng hình ảnh (PNG, SVG).
- Cho phép tải công thức và cấu trúc BDD ở dạng JSON để tái sử dụng.

f. User Interface

- Giao diện web (UI) thân thiện hơn, dễ nhập công thức.
- Có khu vực riêng cho công thức, kết quả BDD, và công cụ điều khiển (zoom, export).
- Handle responsive

g. Error Handling

- Hiển thị thông báo khi công thức không hợp lệ.
- Gợi ý ví dụ cú pháp đúng

4.2 Yêu cầu phi chức năng:

a. Performance (Hiệu năng)

- Ứng dụng xử lý và hiển thị nhanh với công thức ≤ 10 biến.
- Thời gian sinh BDD ≤ 2 giây cho input thông thường.

b. Usability

- Giao diện trực quan, dễ thao tác cho cả người chưa quen với BDD.
- Hỗ trợ hiển thị tooltip hoặc chú thích cho node/cạnh.

c. Scalability

- Hệ thống có thể mở rộng để xử lý công thức phức tạp hơn (≥ 15 biến) trong tương lai.

d. Maintainability

- Code được tách module: parser công thức, BDD generator, visualization.
- Có unit test cho parser và BDD generator.

e. Portability

- Ứng dụng chạy độc lập trên trình duyệt, không yêu cầu cài đặt phần mềm bổ sung.

f. Reliability

- Đảm bảo output BDD luôn chính xác theo công thức input.
- Giảm thiểu lỗi vòng lặp vô hạn khi sinh đồ thị.

5. Các công cụ/công nghệ

Công nghệ sử dụng

- **Frontend:** React hoặc HTML+CSS+JS
- Deploy frontend: Vercel (thông qua Vercel CLI)
- **Backend:** Flask/FastAPI + Python lib dd

Deploy backend: Render

- **Export:** [D3.js](#) cho SVG/PNG JSON.stringify để lưu lại cấu trúc BDD

Collaboration: Github

6. Kết quả đầu ra

a. Sản phẩm phần mềm:

Hoàn thiện một ứng dụng web có khả năng:

- Nhập và xử lý công thức Boolean theo cú pháp chuẩn.
- Tự động xây dựng và biểu diễn Binary Decision Diagram (BDD) tương ứng.
- Hỗ trợ trực quan hóa đồ thị với các thao tác phóng to, thu nhỏ, di chuyển, và làm nổi bật đường đi theo giá trị biến.
- Cho phép xuất đồ thị dưới dạng hình ảnh (PNG, SVG) hoặc mã LaTeX/TikZ để phục vụ tài liệu học thuật.

Ứng dụng chạy ổn định, giao diện trực quan, dễ sử dụng và có thể hoạt động độc lập trên trình duyệt mà không yêu cầu cài đặt phức tạp.

b. Học thuật:

Hoàn thiện báo cáo học thuật mô tả chi tiết:

- Cơ sở lý thuyết về BDD và ROBDD.
- Quy trình và thuật toán chuyển đổi công thức Boolean sang BDD.
- Các kỹ thuật tối ưu và giảm đồ thị để đảm bảo biểu diễn hiệu quả.
- So sánh kết quả với một số công cụ hiện có, đánh giá tính đúng đắn và hiệu quả của hệ thống.

Góp phần làm tài liệu tham khảo cho sinh viên, giảng viên trong việc học tập và giảng dạy các môn liên quan đến logic và cấu trúc dữ liệu.

c. Kỹ năng và kinh nghiệm:

Nhóm sinh viên rèn luyện được khả năng:

- Ứng dụng lý thuyết phức tạp vào hiện thực phần mềm.
- Thiết kế và lập trình hệ thống web tích hợp cả backend (xử lý thuật toán) và frontend (giao diện trực quan).
- Làm việc nhóm, phân chia công việc hợp lý, quản lý tiến độ và sử dụng công cụ hỗ trợ phát triển phần mềm.

Đề tài có thể trở thành nền tảng để tiếp tục phát triển các tính năng nâng cao trong tương lai, như: nhập từ bảng chân trị, so sánh hai công thức Boolean thông qua BDD, hoặc tích hợp vào công cụ kiểm chứng mô hình.