**THUẬT TOÁN LẬP LỊCH DEADLINE MONOTONIC (DM)**

1. **Miêu tả thuật toán Deadline Monotonic**
2. **Mục tiêu** ‑ xác định xem một tập *n* task tuần hoàn/gián đoạn có deadline ràng buộc (*constrained‑deadline, Dᵢ ≤ Tᵢ*) có được lập lịch an toàn trên một CPU đơn hay không, đồng thời cung cấp chính sách lập lịch.
3. **Nguyên lý Deadline Monotonic**
   1. Gán độ ưu tiên tĩnh tỉ lệ nghịch với deadline tương đối *Dᵢ*: deadline càng nhỏ ⇒ ưu tiên càng cao.
   2. Bộ lập lịch runtime luôn chọn job ready có Deadline nhỏ nhất; trưng dụng ngay khi xuất hiện một job có ưu tiên cao hơn.
4. **Điều kiện cần** – tác vụ độc lập, *WCET* đã biết, không share tài nguyên (hoặc dùng mutex với PCP để tránh priority inversion).
5. **Kiểm tra khả năng lập lịch**
   1. **Điều kiện đủ**:
   2. **Phân tích thời gian đáp ứng (RTA)** – lặp công thức cố định:  
       A black and white math symbols

      AI-generated content may be incorrect.  
      *Nếu* Rᵢ converge ≤ Dᵢ *∀ i* ⇒ schedulable.
6. **Tính chất** – DM là thuật toán cố định ưu tiên tối ưu cho tập task có deadline bị chặn: nếu có một chế độ ưu tiên cố định khác lập lịch được thì DM cũng lập lịch được.
7. **Ưu – Nhược điểm của Deadline‑Monotonic**

**Ưu điểm**

* Tối ưu trong lớp lập lịch ưu tiên cố định đối với tác vụ deadline bị chặn.
* Hiệu quả với tải có chu kì dài nhưng deadline ngắn (đảm bảo phản hồi nhanh cho nhiệm vụ khẩn).
* Ổn định hơn RM trong trường hợp overload; các task deadline ngắn (thường quan trọng) vẫn có cơ hội hoàn thành.

**Nhược điểm**

* Phân tích khả năng lập lịch (RTA) phức tạp hơn RM; phải lặp đến hội tụ.
* Cài đặt runtime yêu cầu trưng dụng nhanh → chi phí chuyển ngữ cảnh lớn.
* Khi có deadline > period DM không còn tối ưu; cần thuật toán khác hoặc ưu tiên thủ công.

1. **Miêu tả chương trình cài đặt**

Chương trình mô phỏng DM được viết bằng **C++ 17** trong file DM.cpp.

|  |  |  |  |
| --- | --- | --- | --- |
| **Hàm / Cấu trúc** | **Dữ liệu vào** | **Dữ liệu ra** | **Công việc chính** |
| struct Task | {id, r, C, T, D} | — | Lưu tham số của một tác vụ. r cho phép mỗi task khởi động tại thời điểm tùy ý. |
| struct Job | Tự sinh | — | Thể hiện một instance: id, rem, releaseTime, deadlineTime, Dline. |
| generateJobs(tasks, simTime, jobs) | vector<Task>, int | vector<Job> | Sinh job cho mỗi task: release = r + k·T, deadline = release + D. |
| simulateDM(tasks, simTime) | vector<Task>, int | bool (đủ hạn) + in lịch | Bộ lập lịch DM: luôn chọn công việc có deadline nhỏ nhất, hỗ trợ trưng dụng. |
| responseTimeTest(tasks) | vector<Task> sắp theo D↑ | bool | Phân tích thời gian đáp ứng (RTA) – không phụ thuộc r vì worst‑case synchronous release giả sử tất cả r = 0. |
| main() | — | — | Khởi tạo tập task, chạy RTA, sau đó mô phỏng. |

**2.1 Định dạng dữ liệu I/O**

* A screenshot of a computer

  AI-generated content may be incorrect.**Input (file .txt)**
* **Output terminal**

A screenshot of a computer

AI-generated content may be incorrect.

**2.2 Giải thích chi tiết cấu trúc Task {id, r, C, T, D}**

|  |  |  |
| --- | --- | --- |
| **Trường** | **Ý nghĩa** | **Kiểu dữ liệu** |
| id | Mã định danh duy nhất của task. | int |
| r (*Release Offset*) | Thời điểm phát hành đầu tiên của task; cho phép mô tả khởi động lệch pha. | int |
| C (*WCET*) | Thời gian thực thi tệ nhất của một job. | int |
| T (*Period*) | Khoảng cách giữa hai release liên tiếp. | int |
| D (*Relative Deadline*) | Hạn chót tính từ thời điểm release (yêu cầu D ≤ T). | int |

* 1. **Mã nguồn C++**

[Deadline-Monotonic](https://github.com/adachankawaii/Deadline-Monotonic.git)

**2.4 Tài liệu tham khảo**