|  |  |
| --- | --- |
| TRƯỜNG ĐH GIAO THÔNG VẬN TẢI PHÂN HIỆU TP.HCM  Bộ Môn Công Nghệ Thông Tin | CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM  Độc Lập – Tự Do -Hạnh Phúc |

**ĐỀ CƯƠNG CHI TIẾT ĐỒ ÁN TỐT NGHIỆP**

1. **Thông tin Sinh viên:**

Họ tên : Nguyễn Hoàng Phát Mã sinh viên : 6151071082

Lớp : CQ.61.CNTT Hệ : Chính quy

Ngành đào tạo : Công nghệ thông tin Khoá : 61

Email : nguyenphatbe@gmail.com Số điện thoại : 0817937865

1. **Thông tin Giảng viên hướng dẫn:**

Họ tên : ThS. Trần Phong Nhã Học vị : Thạc sĩ

Email : tpnha@utc2.edu.vn Số điện thoại :

Đơn vị công tác: Trường Đại học Giao thông Vận tải Phân hiệu tại TP. Hồ Chí Minh

**NỘI DUNG**

**I. Tên đề tài**

Tìm hiểu về Bài toán Assignment problem và ứng dụng vào xây dựng website tối ưu hóa

quy trình gán xe vào tuyến.

**II. Giới thiệu**

Trong bối cảnh hiện nay, việc quản lý và điều phối xe tải đối với các kho hàng đang trở nên ngày càng phức tạp và đòi hỏi sự tự động hóa để tối ưu hóa hiệu suất. Điều này đặc biệt quan trọng trong quản lý logistics và vận chuyển, nơi mà việc xác định cách phân bổ tài nguyên một cách hiệu quả có thể giảm thiểu chi phí và tăng cường năng suất.

Bài toán Assignment (gán nhiệm vụ) là một trong những công cụ quan trọng trong lĩnh vực này. Bài toán này đặt ra câu hỏi: “Làm thế nào để phân bổ một tập hợp các tài nguyên (như xe tải) cho một tập hợp các nhiệm vụ (như các tuyến giao hàng) sao cho tổng chi phí và thời gian là tối ưu nhất?”

Trong ngữ cảnh của việc quản lý kho hàng và điều phối xe tải, bài toán này có thể áp dụng để tự động hóa quy trình gán các xe vào các tuyến giao hàng. Thay vì phải phụ thuộc vào quyết định của nhân viên về cách phân bổ xe tải cho mỗi tuyến giao hàng, một hệ thống tự động có thể áp dụng các thuật toán để tối ưu hóa quy trình này. Điều này có thể giảm thiểu thời gian phân bổ, giảm thiểu chi phí nhiên liệu và tăng cường hiệu suất tổng thể của quá trình logistics.

Mục tiêu của đề tài là xây dựng một website tự động hóa quy trình gán xe vào các tuyến giao hàng, sử dụng các thuật toán tối ưu hóa từ bài toán Assignment. Website này sẽ cung cấp một giao diện thân thiện để người dùng có thể nhập thông tin về các tuyến giao hàng cần phục vụ, cùng với thông tin về các xe tải có sẵn. Dựa trên các thông tin này, hệ thống sẽ tự động áp dụng các thuật toán tối ưu hóa để gán các xe vào từng tuyến giao hàng sao cho tổng chi phí và thời gian là nhỏ nhất. Mục tiêu cụ thể bao gồm:

**Tự động hóa quy trình gán xe:** Hệ thống sẽ phát triển một cơ chế tự động để gán các xe tải vào các tuyến giao hàng một cách thông minh, dựa trên thông tin đầu vào về các tuyến giao hàng và bảng giá xe tải có sẵn. Điều này giúp loại bỏ sự phụ thuộc vào quyết định thủ công và giảm thiểu thời gian cần thiết cho quy trình này.

**Tối ưu hóa hiệu suất:** Sử dụng các thuật toán tối ưu hóa từ bài toán Assignment để đảm bảo rằng việc gán xe vào các tuyến giao hàng được thực hiện một cách hiệu quả nhất. Bằng cách này, hệ thống sẽ tối ưu hóa tổng chi phí hoặc thời gian, giúp tăng cường năng suất và lợi nhuận cho doanh nghiệp.

**Giao diện thân thiện người dùng:** Website sẽ được thiết kế với một giao diện thân thiện và dễ sử dụng cho người dùng. Người dùng có thể dễ dàng nhập thông tin về các tuyến giao hàng và bảng giá xe tải, cũng như theo dõi kết quả của quá trình gán một cách trực quan.

**Tích hợp linh hoạt:** Hệ thống sẽ cung cấp khả năng tích hợp linh hoạt với các hệ thống quản lý kho hàng và vận chuyển hiện có. Điều này giúp tự động hóa hoàn toàn quy trình gán xe tải, từ việc nhập thông tin đến việc thực hiện giao hàng, tạo ra một quy trình logistics mạnh mẽ và hiệu quả.

Đề tài này tập trung vào việc tự động hóa quy trình gán xe vào các tuyến giao hàng trong lĩnh vực logistics và vận chuyển. Phạm vi nghiên cứu bao gồm việc phát triển một hệ thống tự động có khả năng nhập thông tin về các tuyến giao hàng và bảng giá xe tải, sau đó áp dụng các thuật toán để tối ưu hóa việc phân bổ xe tải cho từng tuyến giao hàng.

Đối tượng nghiên cứu của đề tài này bao gồm các doanh nghiệp hoạt động trong lĩnh vực logistics và vận tải, đặc biệt là những doanh nghiệp có nhu cầu tự động hóa quy trình điều phối xe tải và tối ưu hóa chi phí logistics. Ngoài ra còn bao gồm người quản lý và quyết định trong các doanh nghiệp logistics, những người có nhu cầu tối ưu hóa quy trình gán xe và tăng cường hiệu suất trong hoạt động vận chuyển và điều phối hàng hóa.

**III. Cơ sở lý thuyết**

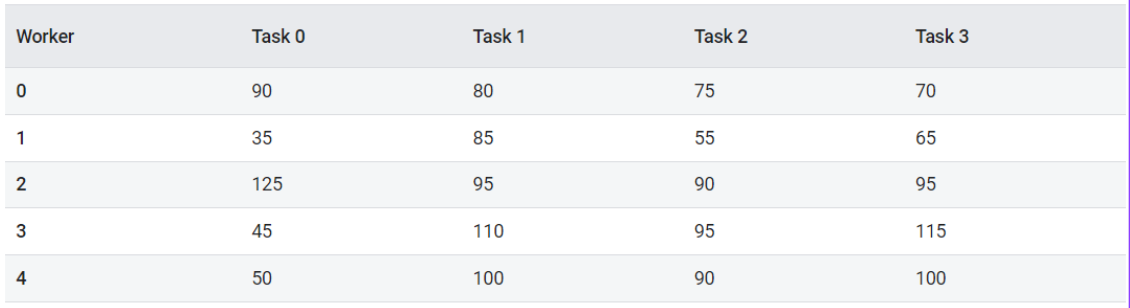
**Bài toán Assignment** là một bài toán quan trọng trong lĩnh vực tối ưu hóa, được áp dụng rộng rãi trong nhiều ngành công nghiệp và lĩnh vực khác nhau. Trong bối cảnh của đề tài, nó trở thành một công cụ mạnh mẽ để tự động hóa quy trình điều phối xe tải vào các tuyến giao hàng trong ngành logistics và vận chuyển.

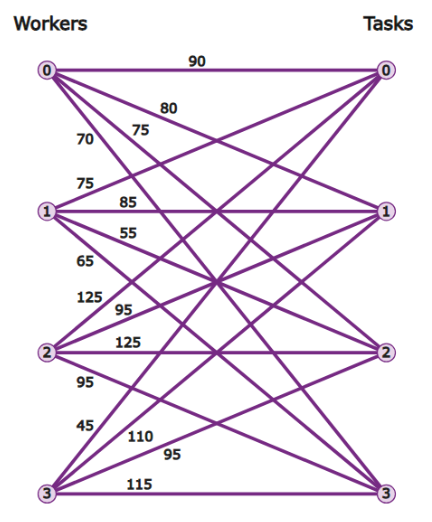
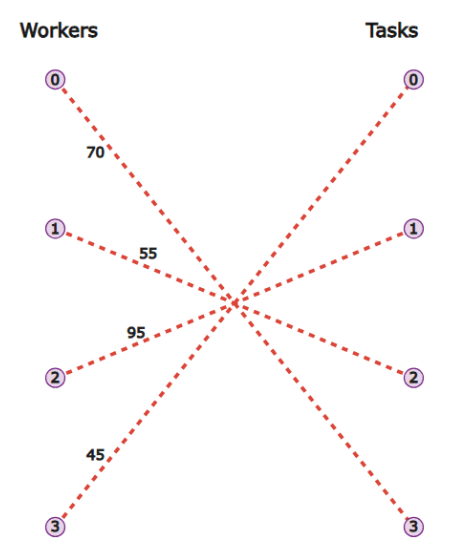
Đối với các doanh nghiệp logistics, việc gán xe tải vào các tuyến giao hàng một cách hiệu quả là một thách thức lớn. Mỗi tuyến giao hàng có thể có các yêu cầu đặc biệt về thời gian, tải trọng, hoặc các ràng buộc khác. Đồng thời, các xe tải có sẵn cũng có các hạn chế về khả năng vận chuyển, tốc độ, và tải trọng. Do đó, việc phân bổ xe tải vào các tuyến giao hàng một cách tối ưu không chỉ giúp giảm thiểu chi phí vận chuyển mà còn đảm bảo thời gian và chất lượng dịch vụ.

Bằng cách áp dụng bài toán Assignment, các doanh nghiệp logistics có thể tự động gán các xe tải vào các tuyến giao hàng sao cho mục tiêu tối ưu nhất được đạt được. Mục tiêu này có thể là giảm thiểu chi phí tổng cộng của quá trình vận chuyển, tối ưu hóa thời gian giao hàng, hoặc cân đối giữa các yếu tố khác nhau như chi phí và thời gian. Điều này giúp doanh nghiệp tăng cường hiệu suất, giảm thiểu lãng phí và tối ưu hóa lợi nhuận.

Ngoài ra, việc tự động hóa quy trình gán xe tải cũng mang lại sự linh hoạt và đáp ứng nhanh chóng hơn đối với biến động trong nhu cầu vận chuyển. Thay vì phải dành nhiều thời gian và công sức để thực hiện các quyết định gán xe tải thủ công, hệ thống tự động có thể đáp ứng tức thì và điều chỉnh quy trình dựa trên thông tin mới và yêu cầu thay đổi từ khách hàng.

Ví dụ: Có 5 worker và 4 tasks, với mỗi tasks có chi phí của từng worker. Vậy bài toán cần giải quyết sẽ tìm ra cách lựa chọn tasks cho mỗi worker sao cho chi phí tối ưu nhất tương ứng với năng lực và tỉ lệ phân bổ.





**Công cụ OR-Tools** là một bộ công cụ mạnh mẽ được phát triển bởi Google Developer dành cho việc giải quyết các bài toán tối ưu hóa. Được viết bằng ngôn ngữ lập trình C++ và cung cấp các giao diện lập trình ứng dụng (API) cho nhiều ngôn ngữ khác nhau như Python, Java, và C#, OR-Tools là một công cụ linh hoạt và tiện ích cho việc giải quyết các vấn đề tối ưu hóa phức tạp.

Với mục tiêu là cung cấp các giải pháp tối ưu cho các bài toán thực tế, OR-Tools cung cấp một loạt các thuật toán và công cụ cho các lĩnh vực như tuyến giao thông, lập kế hoạch sản xuất, quản lý lịch trình, và nhiều vấn đề tối ưu hóa khác. Các thuật toán có sẵn trong OR-Tools bao gồm thuật toán giải bài toán Assignment, tối ưu hóa ràng buộc, tối ưu hóa hàm mục tiêu, và nhiều thuật toán tối ưu hóa khác.

Điều đặc biệt là OR-Tools được thiết kế để có thể tích hợp linh hoạt với các ngôn ngữ lập trình và hệ thống khác nhau. Điều này giúp cho các nhà phát triển có thể sử dụng OR-Tools trong các dự án phần mềm của họ một cách dễ dàng và hiệu quả. Thư viện cung cấp tài liệu hướng dẫn phong phú và ví dụ minh họa đa dạng, giúp người dùng nắm bắt nhanh chóng và áp dụng các tính năng của OR-Tools vào các ứng dụng thực tế của họ.

Với tính linh hoạt, tính mạnh mẽ và khả năng tích hợp cao, OR-Tools là một công cụ quan trọng trong cộng đồng lập trình và nghiên cứu về tối ưu hóa, mang lại giải pháp cho những bài toán phức tạp và đóng góp vào sự phát triển của các ứng dụng thực tiễn và công nghiệp.

**Giải quyết bài toán chi phí sử dụng công cụ OR-Tools:**

Khởi tạo dữ liệu cho bài toán. Bao gồm ma trận chi phí costs, ma trận thời gian times và ma trận thời gian workerShift. Mỗi hàng trong ma trận costs đại diện cho một nhà cung cấp và mỗi cột đại diện cho một công việc. Mỗi hàng trong ma trận times chứa hai giá trị thời gian bắt đầu và kết thúc cho mỗi công việc. Mỗi hàng trong ma trận workerShift chứa hai giá trị thời gian bắt đầu và kết thúc nằm ngoài ca làm việc của nhà cung cấp.

int[][] costs = {

                { 90, 76, 75, 20 },

                { 35, 85, 55, 20 },

                { 35, 85, 55, 20 },

};

int[][] times = { { 0, 10 }, { 11, 20 }, { 19, 30 }, { 19, 30 } };

int[][][] workerShift = {

{ { 0, 4 } }, { { 20, 30 } }, { { 0, 10 }, { 11, 18 } }

};

final int numWorkers = costs.length;

final int numTasks = costs[0].length;

final int[] allWorkers = IntStream.range(0, numWorkers).toArray();

final int[] allTasks = IntStream.range(0, numTasks).toArray();

Tiếp theo khởi tạo một đối tượng CpModel, đại diện cho mô hình của bài toán. Đối tượng CpModel là nơi mà ta định nghĩa các biến, ràng buộc và hàm mục tiêu cho bài toán tối ưu hóa ràng buộc của chúng ta. Bằng cách sử dụng đối tượng này, ta có thể thêm các biến nhị phân (boolean), biến số nguyên, các ràng buộc, và hàm mục tiêu vào mô hình.

Khi một đối tượng CpModel được khởi tạo, ta có thể sử dụng các phương thức của nó để thêm và quản lý các thành phần của bài toán tối ưu hóa ràng buộc, từ đó tạo ra một mô hình hoàn chỉnh để giải quyết bài toán.

CpModel model = new CpModel();

Khởi tạo các biến quyết định. Trong bài toán này, mỗi biến x[worker][task] là một biến nhị phân (boolean) đại diện cho việc công việc task có được giao cho nhà cung cấp worker hay không.

Literal[][] x = new Literal[numWorkers][numTasks];

        for (int worker : allWorkers) {

            for (int task : allTasks) {

x[worker][task] = model.newBoolVar("x[" + worker + "," + task + "]");

            }

        }

Sau khi đã có model và biến, ta sẽ tạo các rảng buộc cho bài toán.

* Ràng buộc về thời gian, đảm bảo thời gian của tuyến phải nằm ngoài khung thời gian của nhà cung cấp không đáp ứng.

// Additional variables and constraints for task scheduling

        for (int worker : allWorkers) {

            List<IntervalVar> lst = new ArrayList<>();

            // shift constraint

            if (workerShift[worker] != null) {

                for (int[] shift : workerShift[worker]) {

                    int shift\_st = shift[0];

                    int shift\_en = shift[1];

                    IntVar \_shift\_st = model.newConstant(shift\_st);

                    IntVar \_shift\_en = model.newConstant(shift\_en);

                    IntVar \_shift\_size = model.newConstant(shift\_en - shift\_st);

                    lst.add(model.newIntervalVar(\_shift\_st, \_shift\_size, \_shift\_en,

                            "worker-task-shift" + worker));

                }

            }

            // time constraint

            for (int task : allTasks) {

                int st = times[task][0];

                int en = times[task][1];

                IntVar \_st = model.newConstant(st);

                IntVar \_en = model.newConstant(en);

                IntVar \_size = model.newConstant(en - st);

                lst.add(model.newOptionalIntervalVar(\_st, \_size, \_en, x[worker][task],

                        "worker-task" + task + worker));

            }

model.addNoOverlap(lst);

}

Cụ thể, vòng lặp đầu tiên for (int worker : allWorkers) duyệt qua tất cả các nhà cung cấp có trong danh sách allWorkers. Trong mỗi vòng lặp này, một danh sách lst được khởi tạo để chứa các biến thời gian của các công việc mà nhà cung cấp đang xem xét.

Trong vòng lặp lồng bên trong for (int[] shift : workerShift[worker]), Mỗi khung thời gian làm việc của nhà cung cấp được xác định bởi thời gian bắt đầu (shift\_st) và thời gian kết thúc (shift\_en) từ mảng workerShift. Ba biến số nguyên \_shift\_st, \_shift\_en, và \_shift\_size được tạo bằng cách sử dụng phương thức model.newConstant() để đại diện cho thời gian bắt đầu, thời gian kết thúc và thời lượng của mỗi khung thời gian làm việc.

Sau đó, tạo biến thời gian cho mỗi khung thời gian làm việc của nhà cung cấp bằng cách sử dụng phương thức model.newIntervalVar(). Mỗi biến thời gian này đại diện cho một khoảng thời gian mà nhà cung cấp không thể thực hiện các công việc.

Tương tự, trong vòng lặp lồng bên trong for (int task : allTasks), mỗi công việc task trong danh sách allTasks được xem xét. Đầu tiên, thời gian bắt đầu (st) và thời gian kết thúc (en) của công việc được lấy từ ma trận times. Ba biến số nguyên \_st, \_en, và \_sten được khởi tạo bằng cách sử dụng phương thức model.newConstant() để đại diện cho thời gian bắt đầu, kết thúc và thời lượng của công việc tương ứng.

Cuối cùng, một biến thời gian tùy chọn (optional interval variable) được tạo bằng cách sử dụng phương thức model.newOptionalIntervalVar(). Biến này chứa thông tin về thời gian bắt đầu (\_st), thời lượng (\_sten), và thời gian kết thúc (\_en) của công việc, cũng như một biến nhị phân x[worker][task] để đại diện cho việc giao công việc task cho nhà cung cấp worker. Chuỗi "worker-task" được sử dụng để gán một tên cho biến.

Sự khác nhau giữa newIntervalVar và newOptionalIntervalVar đó là phương thức model.newIntervalVar() tạo ra các biến thời gian bắt buộc, trong khi model.newOptionalIntervalVar() tạo ra các biến thời gian tùy chọn, cho phép công việc không được lên lịch trong một khung thời gian cụ thể.

Sau khi tạo biến thời gian cho tất cả các công việc và thời gian của nhà cung cấp, ràng buộc model.addNoOverlap(lst) được thêm vào mô hình để đảm bảo rằng mỗi công việc chỉ được lên lịch trong đúng khung thời gian làm việc của nhà cung cấp.

* Ràng buộc về mỗi công việc được giao cho một nhà cung cấp.

for (int task : allTasks) {

            List<Literal> workers = new ArrayList<>();

            for (int worker : allWorkers) {

                workers.add(x[worker][task]);

            }

            model.addExactlyOne(workers);

        }

Cụ thể, vòng lặp bên ngoài for (int task : allTasks) duyệt qua tất cả các công việc có trong danh sách allTasks. Trong mỗi vòng lặp này, một danh sách workers được khởi tạo để chứa tất cả các biến nhị phân (boolean variables) x[worker][task], đại diện cho việc gán công việc task cho các nhà cung cấp khác nhau.

Trong vòng lặp lồng bên trong for (int worker : allWorkers), mỗi nhà cung cấp worker trong danh sách allWorkers được xem xét. Biến x[worker][task] tương ứng với việc gán công việc task cho nhà cung cấp worker. Biến này được thêm vào danh sách workers.

Sau khi đã thêm tất cả các biến x[worker][task] tương ứng với việc gán công việc task cho các nhà cung cấp khác nhau vào danh sách workers, ràng buộc model.addExactlyOne(workers) được thêm vào mô hình. Ràng buộc này đảm bảo rằng chính xác một biến trong danh sách workers sẽ nhận giá trị true, đồng thời các biến còn lại sẽ nhận giá trị false. Điều này đảm bảo rằng mỗi công việc chỉ được giao cho đúng một nhà cung cấp duy nhất.

Sau khi đã có các ràng buộc, ta tiếp tục xác định hàm mục tiêu của bài toán.

// Objective

        LinearExprBuilder obj = LinearExpr.newBuilder();

        for (int worker : allWorkers) {

            for (int task : allTasks) {

                obj.addTerm(x[worker][task], costs[worker][task]);

            }

        }

        model.minimize(obj);

LinearExprBuilder obj = LinearExpr.newBuilder();. Ở đây, một đối tượng LinearExprBuilder được tạo ra để xây dựng hàm mục tiêu tuyến tính. Đối tượng này sẽ được sử dụng để thêm các thành phần của hàm mục tiêu vào.

Vòng lặp hai lớp for duyệt qua tất cả các cặp (nhà cung cấp, công việc) có trong ma trận chi phí costs.

Vòng lặp bên ngoài for (int worker : allWorkers) duyệt qua tất cả các nhà cung cấp.

Vòng lặp lồng bên trong for (int task : allTasks) duyệt qua tất cả các công việc.

Trong mỗi lần lặp của vòng lặp lồng, một hạng tử (term) mới được thêm vào hàm mục tiêu. Hạng tử này được tạo bằng cách sử dụng phương thức addTerm() của đối tượng obj. Hạng tử này có dạng x[worker][task] \* costs[worker][task], trong đó x[worker][task] là biến quyết định đại diện cho việc giao công việc task cho nhà cung cấp worker, và costs[worker][task] là chi phí tương ứng với việc giao công việc đó.

Cuối cùng, hàm mục tiêu được tạo thành là tổng tất cả các hạng tử đã được thêm vào, đại diện cho tổng chi phí của việc gán công việc cho các nhà cung cấp.

model.minimize(obj); hàm mục tiêu được thiết lập bằng cách sử dụng phương thức minimize() của đối tượng mô hình model. Điều này khẳng định rằng mục tiêu của bài toán là tối thiểu hóa giá trị của hàm mục tiêu, tức là tối thiểu hóa tổng chi phí của việc gán công việc cho các nhà cung cấp.

Tiếp theo, ta giải quyết bài toán bằng đối tượng CpSolver.

// Solve

        CpSolver solver = new CpSolver();

        CpSolverStatus status = solver.solve(model);

Cụ thể, tạo một đối tượng CpSolver, đây là một solver được cung cấp bởi công cụ OR-Tools để giải quyết các bài toán tối ưu hóa ràng buộc dựa trên phương pháp lập trình ràng buộc.

CpSolverStatus status = solver.solve(model); Sử dụng solver để giải quyết bài toán được mô hình hóa bởi model. Kết quả của quá trình giải quyết được lưu trữ trong biến status, là một giá trị thuộc kiểu enum CpSolverStatus, biểu thị trạng thái của quá trình giải quyết bài toán.

Nếu status là CpSolverStatus.OPTIMAL, điều này ngụ ý rằng giải pháp tối ưu đã được tìm thấy, tức là đã tìm ra một giải pháp có giá trị của hàm mục tiêu là tối ưu nhất.

Nếu status là CpSolverStatus.FEASIBLE, điều này ngụ ý rằng solver đã tìm thấy một giải pháp khả thi, nhưng không chắc chắn rằng nó là tối ưu nhất.

Nếu không, status có thể là CpSolverStatus.INFEASIBLE, CpSolverStatus.MODEL\_INVALID, CpSolverStatus.UNKNOWN, hoặc các giá trị khác, biểu thị rằng có lỗi trong quá trình giải hoặc mô hình không hợp lệ.

Sau khi quá trình giải quyết bài toán được thực hiện, ta tiếp tục với việc in ra giải pháp đã tìm được (nếu có) và kiểm tra xem giải pháp đó có khả thi hay không.

// Print solution.

        // Check that the problem has a feasible solution.

        if (status == CpSolverStatus.OPTIMAL || status == CpSolverStatus.FEASIBLE) {

            System.out.println("Total cost: " + solver.objectiveValue() + "\n");

            for (int worker : allWorkers) {

                for (int task : allTasks) {

                    if (solver.booleanValue(x[worker][task])) {

                        System.out.println("Worker " + worker + " assigned to task " + task

                                + ".  Cost: " + costs[worker][task]);

                    }

                }

            }

        } else {

            System.err.println("No solution found.");

        }

Trước tiên, một kiểm tra được thực hiện để đảm bảo rằng bài toán có một giải pháp khả thi. Điều này được thực hiện bằng cách kiểm tra xem biến status có giá trị là CpSolverStatus.OPTIMAL hoặc CpSolverStatus.FEASIBLE không. Nếu có ít nhất một giải pháp tối ưu hoặc khả thi được tìm thấy, đoạn mã tiếp tục với việc in ra giải pháp.

Nếu bài toán có một giải pháp tối ưu hoặc khả thi, đoạn mã in ra thông tin về giải pháp đó. Đầu tiên, tổng chi phí của giải pháp được in ra bằng cách sử dụng phương thức solver.objectiveValue(), đại diện cho giá trị của hàm mục tiêu tối ưu. Sau đó, vòng lặp duyệt qua tất cả các cặp (nhà cung cấp, công việc) và kiểm tra xem biến x[worker][task] có được gán giá trị true không. Nếu có, thông tin về việc gán công việc đó cho nhà cung cấp tương ứng được in ra, bao gồm thông tin về công việc, nhà cung cấp và chi phí tương ứng.

Nếu không có giải pháp nào được tìm thấy, một thông báo lỗi được in ra để thông báo rằng không có giải pháp nào được tìm thấy cho bài toán đã cho.

Qua đó, đoạn mã này không chỉ giúp hiển thị thông tin về giải pháp mà còn giúp xác nhận tính khả thi của giải pháp, đảm bảo rằng kết quả được in ra là đáng tin cậy.

Kết quả của quá trình giải quyết bài toán được hiển thị như sau:

Total cost: 195.0

Worker 0 assigned to task 3. Cost: 20

Worker 1 assigned to task 0. Cost: 35

Worker 1 assigned to task 1. Cost: 85

Worker 2 assigned to task 2. Cost: 55

Điều này có nghĩa là bài toán đã được giải quyết thành công và đã tìm thấy một giải pháp khả thi. Giải pháp này gồm các phân công công việc cho các nhà cung cấp, với tổng chi phí là 195.0.

Cụ thể:

Nhà cung cấp 0 được giao công việc 3 với chi phí là 20.

Nhà cung cấp 1 được giao công việc 0 với chi phí là 35.

Nhà cung cấp 1 được giao công việc 2 với chi phí là 85.

Nhà cung cấp 2 được giao công việc 2 với chi phí là 55.

Các nghiên cứu, bài báo, tài liệu tham khảo có liên quan đến đề tài:

The Assignment Problem and Its Relation to Logistics Problems - [*https://www.mdpi.com/1999-4893/15/10/377*](https://www.mdpi.com/1999-4893/15/10/377)

Solving an Assignment Problem -[*https://developers.google.com/optimization/assignment/assignment\_example*](https://developers.google.com/optimization/assignment/assignment_example)

**IV. Phương pháp nghiên cứu**

**Thu thập dữ liệu từ nguồn nội bộ:** Dữ liệu về các tuyến giao hàng và thông tin về các xe tải có thể được trích xuất từ hệ thống quản lý kho hàng và vận chuyển của doanh nghiệp logistics. Các thông tin này sẽ cung cấp cơ sở dữ liệu ban đầu để áp dụng thuật toán.

**Tiền xử lý dữ liệu:** Dữ liệu thu thập được cần phải được tiền xử lý để chuẩn hóa và làm sạch trước khi đưa vào thuật toán. Điều này bao gồm việc xử lý dữ liệu thiếu, loại bỏ dữ liệu nhiễu, và chuyển đổi dữ liệu vào định dạng phù hợp để sử dụng với thư viện OR-Tools.

**Áp dụng thuật toán OR-Tools:** Sử dụng công cụ OR-Tools của Google để áp dụng các thuật toán tối ưu hóa vào quy trình gán xe vào các tuyến giao hàng. Công cụ này cung cấp các giải pháp mạnh mẽ như thuật toán gán (Assignment algorithm), giúp tối ưu hóa việc phân bổ tài nguyên.

**Đánh giá và kiểm tra hiệu suất:** Sau khi có kết quả từ thuật toán OR-Tools, cần tiến hành đánh giá và kiểm tra hiệu suất của phân bổ được tạo ra. Điều này bao gồm việc đánh giá các tiêu chí như chi phí tổng cộng, thời gian vận chuyển, hoặc hiệu quả sử dụng tài nguyên. Các kết quả được so sánh với các phân bổ khác hoặc với các tiêu chuẩn hoạt động đã được đề ra để đánh giá tính hiệu quả của hệ thống.

**V. Kết quả dự kiến**

Dựa trên các phương pháp nghiên cứu và sử dụng công cụ OR-Tools của Google, dự kiến đề tài sẽ đạt được những kết quả đáng chú ý sau:

Hệ thống sẽ được phát triển với khả năng tự động gán các xe tải vào các tuyến giao hàng, dựa trên thông tin nhập vào về các tuyến giao hàng và xe tải. Sử dụng thuật toán tối ưu hóa, kỳ vọng rằng hệ thống sẽ tạo ra các phân bổ tối ưu, giảm thiểu chi phí và thời gian.

Giao diện người dùng trên website sẽ được xây dựng một cách thân thiện và dễ sử dụng. Người dùng sẽ có thể nhập thông tin và theo dõi kết quả của quá trình gán xe vào các tuyến giao hàng một cách trực quan. Đồng thời, hệ thống cũng sẽ cung cấp các biểu đồ và báo cáo để hỗ trợ quyết định quản lý.

Kết quả của hệ thống sẽ được đánh giá thông qua các thử nghiệm và so sánh với các tiêu chuẩn hoạt động đã được đề ra. Mục tiêu là tạo ra các phân bổ hiệu quả và đáng tin cậy trong điều kiện thực tế, giúp cải thiện hiệu suất và lợi nhuận trong hoạt động vận chuyển và điều phối hàng hóa.

Cuối cùng, hệ thống sẽ được thiết kế để có khả năng tích hợp linh hoạt với các hệ thống quản lý kho hàng và vận chuyển hiện có. Điều này sẽ tạo ra một quy trình logistics linh hoạt và hiệu quả, giảm thiểu sự phụ thuộc vào quyết định thủ công và tăng cường năng suất tổng thể.

**VI. Đóng góp của đề tài**

Đề tài này đem lại những đóng góp quan trọng đối với cả lĩnh vực nghiên cứu và thực tiễn trong lĩnh vực logistics và vận chuyển. Trong lĩnh vực nghiên cứu, đề tài giúp mở ra cánh cửa cho việc khai thác và ứng dụng của bài toán Assignment vào các quy trình vận chuyển và điều phối hàng hóa. Việc áp dụng các phương pháp tối ưu hóa mới trong thực tế mở ra tiềm năng cho việc phát triển các thuật toán và giải pháp sáng tạo hơn trong tương lai.

Tích hợp công nghệ vào thực tiễn là một phần quan trọng của đề tài này. Hệ thống tự động hóa được phát triển trong đề tài không chỉ là một công cụ mạnh mẽ để tối ưu hóa quy trình gán xe vào các tuyến giao hàng mà còn là một bước tiến quan trọng trong việc thúc đẩy sự hiện đại hóa trong ngành logistics. Việc sử dụng công nghệ giúp tăng cường hiệu suất và giảm thiểu chi phí, đồng thời mang lại lợi ích kinh tế đáng kể cho các doanh nghiệp.

Ngoài ra, đề tài giải quyết một thách thức thực tiễn đối với các doanh nghiệp logistics đó là việc tự động hóa quy trình gán xe vào các tuyến giao hàng. Thay vì dựa vào quyết định thủ công, hệ thống tự động này mang lại sự linh hoạt và hiệu quả cao hơn trong quản lý và điều phối hàng hóa, giúp giải quyết các vấn đề thực tế một cách hiệu quả và tự động hóa hóa toàn bộ quy trình logistics.

Như vậy, đề tài này không chỉ đem lại sự cải thiện về hiệu suất và hiệu quả trong lĩnh vực logistics mà còn đóng góp vào việc thúc đẩy sự phát triển và ứng dụng của công nghệ trong ngành.

**VII. Cấu trúc đồ án**

**CHƯƠNG 1: Tổng quan**

**1.1. Tình hình nghiên cứu**  
**1.2 Lý do chọn đề tài**

**1.3 Mục tiêu, nội dung, phương pháp nghiên cứu đề tài**

1.3.1 Mục tiêu đề tài

1.3.2 Nội dung đề tài

1.3.3 Phương pháp nghiên cứu đề tài

**1.4 Đối tượng và phạm vi đề tài**

**CHƯƠNG 2: Cơ sở lý thuyết**

* 1. **Bài toán Assignment**
  2. **Thư viện Or-Tools**
  3. **Công nghệ Spring Boot**

**CHƯƠNG 3: Phân tích hệ thống**

**CHƯƠNG 4: Thiết kế giao diện**

**Kết luận và Kiến nghị**

**Kết quả đạt được**

**Hạn chế**

**Hướng phát triển**

**Tài liệu tham khảo**

**VIII. Tài liệu tham khảo**

Supply chain là gì, vai trò và các hoạt động trong supply chain - [*https://www.pace.edu.vn/tin-kho-tri-thuc/supply-chain-la-gi*](https://www.pace.edu.vn/tin-kho-tri-thuc/supply-chain-la-gi)

The Assignment Problem and Its Relation to Logistics Problems - [*https://www.mdpi.com/1999-4893/15/10/377*](https://www.mdpi.com/1999-4893/15/10/377)

Solving an Assignment Problem -[*https://developers.google.com/optimization/assignment/assignment\_example*](https://developers.google.com/optimization/assignment/assignment_example)

**IX. Kế Hoạch thực hiện và tiến độ nghiên cứu**

Thời gian và nội dung công công việc theo tuần.

|  |  |  |
| --- | --- | --- |
| **Thời gian** | **Nội dung công việc** | **Ghi chú** |
| Tuần 1 (18/03 - 25/03) | Chọn đề tài, xây dựng đề cương. |  |
| Tuần 2 – 3  (25/03 – 08/04) | Tìm hiểu về bài toán Assignment problem, các vấn đề và ứng dụng phổ biến.  Đọc và tìm hiểu các tài liệu tham khảo cơ bản về Assignment problem. |  |
| Tuần 4 (08/04 – 14/04) | Tìm hiểu về công cụ và thư viện phổ biến về Assignment problem. |  |
| Tuần 5 – 6  (15/04 – 28/04) | Thu thập dữ liệu liên quan đến bài toán.  Tiền xử lý dữ liệu: làm sạch, chuẩn hóa, và biểu diễn dữ liệu cho phù hợp với mục tiêu nghiên cứu. Thử nghiệm các trường hợp. |  |
| Tuần 7 – 8  (29/04 – 12/05) | Phân tích kết quả từ các thử nghiệm và đánh giá hiệu suất của phương pháp.  Xác định chiến lược cụ thể cho việc phát triển tiếp theo của dự án. |  |
| Tuần 9 – 10  (13/05 -26/05) | Phát triển website áp dụng phương pháp Assignment hiệu quả nhất. |  |
| Tuần 11 – 12  (27/05 – 09/06) | Tổng hợp kết quả và viết báo cáo về đồ án, bao gồm mô tả vấn đề, phương pháp, kết quả và đánh giá.  Chuẩn bị bài thuyết trình để trình bày kết quả. |  |
| Tuần 13 (10/06 – 16/06) | Sửa đổi và hoàn thiện báo cáo dự án dựa trên phản hồi từ giáo viên hướng dẫn.  Chuẩn bị bài thuyết trình cuối cùng và thực hiện thử nghiệm. |  |
| Tuần 14 (15/06 – 22/06) | Duyệt đồ án |  |
| Tuần 15 (22/06 – 28/06) | Bảo vệ đồ án |  |

|  |  |  |
| --- | --- | --- |
|  |  | ……ngày….tháng….năm…. |
| **Trưởng Bộ Môn** | **Ý kiến của GVHD** | **Sinh viên thực hiện** |
| **ThS. Trần Phong Nhã** | **ThS. Trần Phong Nhã** | **Nguyễn Hoàng Phát** |