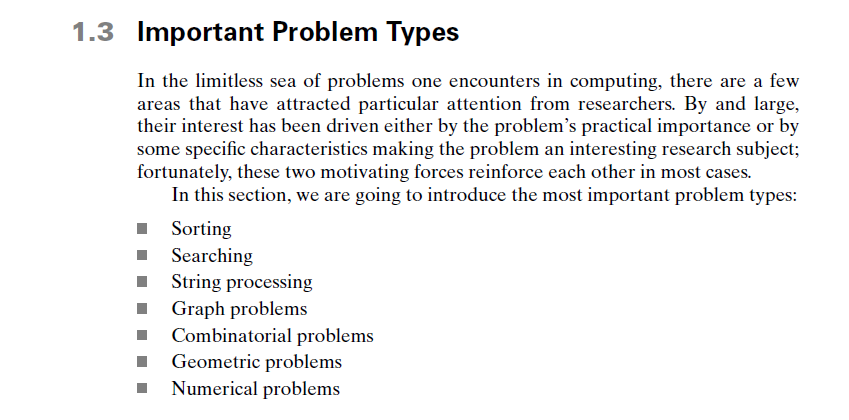
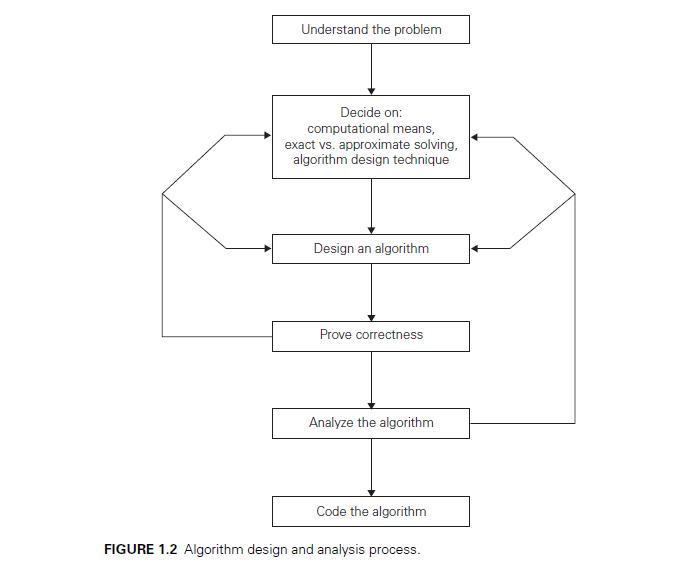
**Chương 1: Tổng quan về vấn đề/bài toán và thuật toán**

* 1. **Vấn đề và biểu diễn vấn đề**
* **Khái niệm vấn đề/bài toán (**[**problem**](file:///C:\Users\noka\Dropbox\Reference\1,%20problem.ppt)**) và ví dụ**.  
  Lưu ý phân biệt:   
  + Bài toán tổng quát (lớp bài toán) và bài toán cụ thể  
  + Bài toán thực tiễn và bài toán dạng mô hình hóa
* **Các thành phần của vấn đề**:   
  (1) **giả thiết (input)**,   
  (2) **mục tiêu**,   
  (3) các **điều kiện ràng buộc hay phạm vi**,   
  (4) **cơ sở dữ liệu – thông tin – tri thức** của vấn đề.
* **Biểu diễn (hay mô hình hóa) vấn đề, ký hiệu:**
* Vấn đề phải được đặc tả hay mô hình hóa dựa trên công cụ toán học hay các ngôn ngữ đặc tả (hình thức) cho máy tính.
* Mô hình cho từng thành phần và tổng hợp lại 🡪 mô hình cho vấn đề.
* Các ví dụ: giải pt bậc 2, tính toán ma trận, tính toán đa thức, tìm đường đi, sắp thời khoa biểu, sắp xếp và tìm kiếm, mã hóa, nén dữ liệu, …
* Tham khảo: [[1](file:///C:\Users\DayHoc\Algorithm-ProblemSolving\problem.ppt)], [[2](file:///C:\Users\DayHoc\Algorithm-ProblemSolving\problem%20solving_reasoning_07.ppt)].
  1. **Phân loại vấn đề:**



* 1. **Tiếp cận thuật toán cho giải quyết vấn đề.**
* Nhắc lại khái niệm thuật toán và vai trò của nó, biểu diễn thuật toán và ví dụ.  
  [[1](file:///C:\Users\DayHoc\Algorithm-ProblemSolving\1-Algorithm%20fs.ppt)], [[2](file:///C:\Users\DayHoc\Algorithm-ProblemSolving\2-ch02n-Algorithm%20Analysis%20(Fundamental).ppt)]
* Giải quyết vấn đề dựa trên việc thiết kế thuật toán cho nó.



* Phân loại thuật toán: ([xem](file:///C:\\Users\\noka\\Dropbox\\Reference\\35-algorithm-types.ppt))
  1. Độ phức tạp của thuật toán và các tính chất

([xem](file:///C:\\Users\\noka\\Dropbox\\Reference\\2b,%20ch02n%20-%20Fundamentals%20of%20the%20Analysis%20of%20Algorithm%20Efficiency.ppt))

* 1. Hiệu chỉnh, cải tiến nâng cao hiệu quả thuật toán
  2. Các ví dụ (case study) và bài tập

**TÓM TẮT**

TỔNG QUAN VỀ VẤN ĐỀ VÀ THUẬT TOÁN

* **Khái niệm vấn đề:**
* Problem: a matter or situation regarded as unwelcome or harmful and needing to be dealt with and overcome.
  + - a thing that is difficult to achieve.
    - An inquiry starting from given conditions to investigate or demonstrate a fact, result, or law.
* Define a problem:
* Tình huống, ngữ cảnh, cơ sở dữ liệu-thông tin-tri thức.
* Giả thiết (input)
* Mục tiêu, yêu cầu (output)
* Điều kiện, ràng buộc, phạm vi.

**Phát biểu (define) vấn đề cho các bài toán sau đây:**

1. Tìm đường đi trên mạng giao thông TPHCM, sử dụng xe gắn máy 2 bánh

* Căn cứ: CSDL về mạng giao thông TPHCM gồm các đường đi (tên), giao lộ, chiều và độ dài, độ rộng, v.v…
* Giả thiết: điểm xuất phát (giao lộ) a, điểm cần tới z
* Kết quả: đường đi ngắn nhất từ a đến z
* Điều kiện, giới hạn ?

1. Tìm đường đi ở TPHCM bằng BUS.

* Có trước: CSDL về các tuyến xe BUS gồm mã số xe BUS và lộ trình tương ứng, v.v…
* Input: trạm đầu, trạm cuối
* Output: đường đi bằng BUS

1. Sắp TKB trường học.
2. Tìm tài liệu trong kho tài nguyên học tập dựa trên chủ đề.

Ứng dụng: …

1. Tìm luật liên hệ giữa các thuộc tính từ một bảng CSDL.
2. Tìm lời giải cho bài toán suy diễn dựa trên cơ sở (tri thức) Hệ luật dẫn.  
   ứng dụng: …
3. Giảm khâu nhập liệu của phòng xét nghiệm vào phần mềm đang có sẵn.
4. Xác định giới tính qua tên (đầu đủ)
5. Phân loại nhận xét đánh giá của người dùng đối với các điện thoại di động.

* Đặc tả vấn đề, sử dụng công cụ toán học? và tin học?
* **Khái niệm về thuật toán, thuật giải (algorithm):**
* In [mathematics](http://en.wikipedia.org/wiki/Mathematics) and [computer science](http://en.wikipedia.org/wiki/Computer_science), an **algorithm** is a step-by-step procedure for calculations. Algorithms are used for [calculation](http://en.wikipedia.org/wiki/Calculation), [data processing](http://en.wikipedia.org/wiki/Data_processing), and [automated reasoning](http://en.wikipedia.org/wiki/Automated_reasoning).
* Representation of algorithms: **pseudo-code**.
* Properties:
* Xác định
* Hữu hạn
* Đúng
* Hiệu quả: (1) Thời gian, (2) không gian
* Complexity of algorithms and notations:
* Cơ sở tính toán: Số lượng thao tác cơ bản, là hàm f(n: input data size).
* O(.), Θ(.)
* Cần dùng công cụ toán học cho tính toán độ phức tạp.
* **Tiếp cận thuật toán giải quyết vấn đề:**
* Giải quyết vấn đề bằng cách xây dựng thuật toán (thuật giải) cho vấn đề.
* Các bước thực hiện:
  + Xác định vấn đề và biểu diễn (mô hình hóa) vấn đề🡪Mô hình của vấn đề.
  + Xây dựng thuật toán (thuật giải)  
    🡪thuật giải cho vấn đề (dạng mã giả)
  + Chứng minh tính đúng đắn. 🡪 đáp ứng yêu cầu đặt ra.
  + Phân tích thuật toán (xét tính hiệu quả): lý thuyết, thực nghiệm🡪 khẳng định về tính hiệu quả của của thuật toán.  
    + Lý thuyết: tính toán độ phức tạp của thuật giải 🡪Lớp độ phức tạp.  
    + Thực nghiệm: chọn dữ liệu bảo đảm tin cậy, thử thuật giải trên dữ liệu mẫu, thống kê và đánh giá kết quả thử nghiệm.
  + Cải tiến, nâng cao hiệu quả của thuật toán
  + Cài đặt ứng dụng và thử nghiệm.

**BÀI TẬP:**

Phát biểu vấn đề và xây dựng giải pháp cho một số vấn đề trong lý thuyết đồ thị:

1. Tìm đường đi ngắn nhất

* Căn cứ: Đồ thị G = (V, E) vô hướng có trọng số
* Input: đỉnh xuất phát a, đỉnh mục tiêu z
* Output: Đường đi P từ a đến z (trên đồ thị G)
* Điều kiện: P có độ dài ngắn nhất (so với …)

1. Tìm chu trình Euler
2. Tìm chu trình Hamilton
3. Tìm MST (Minimal Spanning Tree)
4. Khảo sát sự đẳng cấu của 2 đồ thị

* Căn cứ:
* Input: G1 = (V1, E1) và G2 = (V2, E2)
* Output: kết quả Yes/No;  
  Trường hợp Yes, thì cho biết song ánh f : V1 🡪 V2.
* Điều kiện: G1 và G2 đơn giản, vô hướng

Bước 1:

S = Bijection(V1, V2);

Bước 2:

For f  S do

2.1: xây dựng ánh xạ ~f từ E1 vào E2 bởi  
 ~f (a,b) = ( f(a) , f(b) )

2.2: if ( ~f là song ánh )  
 Dừng và KL: có sự đẳng cấu

Bước 3:

KL trường hợp không có đẳng cấu

k=0, … n-1

tk = | Wk |

n = t0 + t1+ … + tn-1

Số lần lập duyệt song ánh là: **t0! t1! … tn-1!**  
 n=10

4 đỉnh bậc 2,

4 đỉnh bậc 3

2 đỉnh bậc 4

10! So với T = 4!4!2!

Phát biểu vấn đề và xây dựng giải pháp cho một số vấn đề ứng dụng:

1. Tìm đường đi trên mạng giao thông
2. Tìm đường đi xe BUS ở TPHCM
3. Sắp thời khóa biểu

**Bài tập: phát biểu và định nghĩa bài toán (hay vấn đề)**

1. Sắp TKB trường học

* Căn cứ: Các môn học, GV và khả năng dạy chuyên môn, thông tin phòng học
* Input: các môn sẽ mở trong học kỳ, DS GV tương ứng cho các môn
* Output: TKB phù hợp.
* Phạm vi: trường đại học theo tín chỉ cấp trường.
* các ràng buộc  
  + Cứng   
  + Mềm

1. Tìm tài liệu trong kho tài nguyên học tập dựa trên chủ đề.

* Căn cứ:  
  + Kho tài nguyên học tập trên máy tính.  
  + Chủ đề và biểu diễn 🡪 danh mục chủ đề
* Input: một chủ đề
* Output: danh sách tài liệu phù hợp với chủ đề
* Phạm vi: tài liệu về KHMT.

BÀI TOÁN PHÂN CÔNG & CÁC DẠNG MỞ RỘNG:

Có n việc, m máy (người); biết thời gian máy i thực hiện việc j là t[i][j]. Tìm phương án phân công tốt nhất.

BÀI TOÁN TÌM ĐƯỜNG ĐI (NGẮN NHẤT) TRÊN MẠNG GIAO THÔNG (tphcm):

**Chương 2: Cơ sở toán học**

2.1 Tập hợp, quan hệ, hàm và ánh xạ

* [Tham khảo 1](file:///E:\DayHoc\Algorithm-ProblemSolving\Set-Relation-Function\Ch2%20-%20Set-relation-function-sequence.pdf).

2.2 Quan hệ tương đương và thứ tự

* [Tham khảo 1](file:///E:\DayHoc\Algorithm-ProblemSolving\Set-Relation-Function\Ch2%20-%20Set-relation-function-sequence.pdf).
* [Tham khảo thêm về quan hệthứ tự](file:///E:\DayHoc\Algorithm-ProblemSolving\Set-Relation-Function\ch8.6%20-%20Partial%20orderings.ppt).
* Tham khảo thêm về quan hệ tương đương.

Ví dụ 1: Mô hình hóa vấn đề (bài toán)

* Dựa trên các công thức hay định lý liên hệ tính toán trên các yếu tố của một tam giác, hãy tính một số yếu tố từ một số yếu tố được cho trước giá trị.
* Căn cứ có trước: các công thức hay định lý liên hệ tính toán trên các yếu tố của một tam giác.  
  + các yêu tố và ký hiệu  
  + các công thức tính toán
* Input:   
  1) tập X ⊆ M và mỗi x∈X có giá trị cho trước cụ thể.  
  2) tập Y ⊆ M gồm những yếu tố cần tìm (tính) giá trị
* Output: giá trị cụ thể của mỗi y∈Y

Mô hình toán học cho vần đề (bài toán) trên như sau:

<về nhà>

2.3 Toán tử (phép toán) và các cấu trúc đại số

* Phép toán hai ngôi và các tính chất. [[1](file:///E:\DayHoc\Algorithm-ProblemSolving\Operators%20and%20algebraic%20structure\nat_bin_op.pdf)], [[2](file:///E:\DayHoc\Algorithm-ProblemSolving\Operators%20and%20algebraic%20structure\Sach%20Dai%20So%20Dai%20Cuong.pdf)]
* Cấu trúc nhóm và các đồng cấu. [[1](file:///E:\DayHoc\Algorithm-ProblemSolving\Operators%20and%20algebraic%20structure\III.Group.ppt)], [[2](file:///E:\DayHoc\Algorithm-ProblemSolving\Operators%20and%20algebraic%20structure\Sach%20Dai%20So%20Dai%20Cuong.pdf)]
* Vành, trường. [[2](file:///E:\DayHoc\Algorithm-ProblemSolving\Operators%20and%20algebraic%20structure\Sach%20Dai%20So%20Dai%20Cuong.pdf)]

1. Giải bài toán tính toán trên tam giác: Trong 1 tam giác ABC cho trước giá trị của một vài yếu tố (của tam giác), yêu cầu tính giá trị của một số yếu tố được quan tâm.

Mẫu 1: cho a = 5, b = 4, A = pi/2; tính S, R

Mẫu 2: cho a = 5, A = pi/2, B = pi/3; tính b, c

Mô tả problem:

* Căn cứ có trước gồm:   
  + các yếu tố của tam giác ABC: 3 cạnh a, b, c, … lấy giá trị thuộc R+ (tập các số thực dương)  
  + Các công thức đã biết: A+B+C = pi; 2\*p =a+b+c;   
   a / sin (A) = b/ sin(B) = c /sin(C) = 2\*R;  
   a^2 = b^2 + c^2 -2\*b\*c\*cos(A);  
   S = a \* ha / 2; S = a\*b\*sin(C) / 2; …
* Input:  
  + Các yếu tố với giá trị cho trước.  
  + Các yếu tố cần tính
* Output:  
  + giá trị kết quả của các yếu tố cần tính  
  + Quá trình dùng các công thức để tính ra kết quả

Mô hình hóa (dùng cơ sở toán học):

* Căn cứ:  
  + tập M gồm các biến thực dương:  
   M = { a, b, c, A, B, C, S, p, R, ha, hb, hc, … }  
  + tập các công thức dạng đẳng thức đã được chứng minh:  
   F = { A+B+C = pi, 2\*p=a+b+c, … }
* Input:  
  + GT = { <biến> = <giá trị> | <biến> lấy trong M }  
  + KL ⊆ M
* Output:  
  + KQ = { <biến> = <giá trị> | <biến> thuộc KL }

Tiếp …

* 1. Logic toán học: cơ sở khoa học cho lập luận và chứng minh, biểu diễn (mô hình hóa), tính toán và lập trình cho máy tính

Khái quát: các ký hiệu, diễn dịch và ngữ nghĩa, toán tử logic, các biểu thức (công thức logic), các tiên đề và các luật.

* Logic Mệnh đề
* Logic vị từ
* Logic đa trị, Logic mờ
* Logic thời gian (thời điểm, thời khoảng)
* Xem [[3](file:///C:\Users\noka\Dropbox\Logic%20-%20ung%20dung%20trong%20Tin%20hoc\Predicate%20Logic)]
* Xem [[4](file:///C:\Users\noka\Dropbox\Maths%20for%20Computer%20Science\Fuzzy%20Logic)]
  1. Phương pháp đếm

2.6 Đồ thị

* Definition, representation [[5](file:///C:\Users\noka\Dropbox\Onthi-CaoHoc+DeThi\Do%20thi\9_introduction-graph-theory.ppt),[6](file:///C:\Users\noka\Dropbox\Onthi-CaoHoc+DeThi\Do%20thi\Estrada_tutorial1.ppt),[7](file:///C:\Users\noka\Dropbox\Onthi-CaoHoc+DeThi\Do%20thi\lecture%201.ppt)]
* Problems and algorithms
* Isomorphism

2.7 Toán cao cấp:

\* Giải tích (1 biến, nhiều biến)

\* Đại số tuyến tính

2.7Lý thuyết chuỗi và hàm sinh

2.8Nhóm hoán vị và nghịch thế

(xem [1](file:///C:\Users\DayHoc\Phuong%20phap%20Toan%20cho%20Tin%20hoc\Chuoi%20luy%20thua%20&%20ham%20sinh,%2029-05-2005.doc), [2](file:///C:\Users\DayHoc\Phuong%20phap%20Toan%20cho%20Tin%20hoc\2008-04,%20Phan%20tich%20mot%20so%20thuat%20toan.doc))

2.9Xác suất và thống kê

MỘT SỐ KẾT QUẢ ĐÁNG NHỚ:

* Hàm  là hàm sinh của dãy:

 , *n* = 1, 2, …



Dãy {*Hn*}được gọi là***dãy số điều hòa***, đóng vai trò quan trọng trong số học cũng như trong việc phân tích độ phức tạp của một số thuật toán.

* **Công thức quan trọng về *Hn***:

****

trong đó hằng sốγ (được gọi là***hằng số Euler***) tính theo công thức giới hạn



Ta nhận thấy khi  nghĩa là

**Hn ~ Ln(n)**

Với n=6 ta có một khai triển đến cấp 6 theo 



≈ Ln(n) + 0.577

* **Công thức Stirling, tính chất của giai thừa**:

Người ta chứng minh được rằng giới hạn sau đây tồn tại hữu hạn vàđược gọi là***hằng số Stirling***:



Ta cũng có:



và



Dùng tích phân của hàm biến phức ta có thể chứng minh rằng:



vàđược ***công thức Stirling*** sau đây:



Suy ra



## Một số tính chất khác của giai thừa:

1. Giai thừa có tốc độ tăng nhanh hơn [hàm mũ](http://vi.wikipedia.org/wiki/H%C3%A0m_m%C5%A9) nhưng chậm hơn [hàm mũ hai tầng](http://vi.wikipedia.org/w/index.php?title=H%C3%A0m_m%C5%A9_hai_t%E1%BA%A7ng&action=edit&redlink=1) (abc) có cùng cơ số và mũ.
2. \log n! = \sum_{x=1}^n \log x.
3.  \int_1^n \log x \, dx \leq \sum_{x=1}^n \log x \leq \int_0^n \log (x+1) \, dx
4.  n\log\left(\frac{n}{e}\right)+1 \leq \log n! \leq (n+1)\log\left( \frac{n+1}{e} \right) + 1.
5. e\left(\frac ne\right)^n \leq n! \leq e\left(\frac{n+1}e\right)^{n+1}.
6. n!\approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n.(Công thức [Stirling](http://vi.wikipedia.org/w/index.php?title=Stirling&action=edit&redlink=1)).
7. n! > \sqrt{2\pi n}\left(\frac{n}{e}\right)^n.
8. \log n! \approx n\log n - n + \frac {\log(n(1+4n(1+2n)))} {6} + \frac {\log(\pi)} {2}.

* **Công thức Gamma, giai thừa của số thực**:

### Công thức Gamma[[sửa](http://vi.wikipedia.org/w/index.php?title=Giai_th%E1%BB%ABa&action=edit&section=5" \o "Sửa đổi phần )]

Là công thức mang tên một [chữ cái Hy Lạp](http://vi.wikipedia.org/wiki/B%E1%BA%A3ng_ch%E1%BB%AF_c%C3%A1i_Hy_L%E1%BA%A1p) do nhà toán học [Pháp](http://vi.wikipedia.org/wiki/Ph%C3%A1p), [Adrien-Marie Legendre](http://vi.wikipedia.org/wiki/Adrien-Marie_Legendre) đề ra. Hàm số này có dạng sau:

\Gamma(z) = \int_0^\infty  t^{z-1} e^{-t}\,{\rm d}t

Bằng phương pháp tích phân từng phần ta có được:

\Gamma(z+1)=z \, \Gamma(z)\,.

Khi đó ta có:

z! = \Gamma(z + 1).\,

* Recursion, recurrences, and Induction:   
  Tham khảo:

[1] Stein, Drysdale, Bogart. ***Discrete Mathematics for Computer Scientists***. Addison-Wesley, 2011.

[2] Kenneth H. Rosen, ***Discrete Mathematics and Its Applications***6/e, McGraw – Hill, 2007.

\* [Xem thêm](file:///C:\Users\DayHoc\Algorithm-ProblemSolving\ThuatToan-DoPhucTap).

**Theorem 4.1:**

If T(n) = rT(n-1)+a, T(0) = b, and r ≠ 1, then

T(n) = rnb + a.(1-rn)/(1-r)

for all nonnegative integers n.

**Theorem 4.4:**

Let r be a nonnegative quantity whose value is independent of n and not equal to 1. Let t(n) be the largest term of the geometric series .

Then the value of the geometric series is Θ(t(n)).

**Def:** A recurrence of the form

T(n) = f(n)T(n-1) + g(n)

Is called a **first-order linear recurrence**

**Theorem 4.5:**

For any positive constants a and r and any function g defined on the nonnegative integers, the solution to the first-order linear recurrence



is T(n) = rna + 

**Lemma:**

Suppose that we have a recurrence of the form

T(n) = aT(n/2) + n,

where a is a positive integer and T(1) is nonnegetive. Then we have the following big Θ bounds on the solution:

* + 1. If a < 2, then T(n) = Θ(n).
    2. If a = 2, then T(n) = Θ(n log n).
    3. If a > 2, then T(n) = Θ.

**Theorem 4.9: (Master theorem)**

Let a and b be positive real numbers, with a ≥ 1 and b > 1.Let T(n) be defined for integers n that are powers of b by



Then we have the following:

1. If f(n) = Θ(nc), where logba < c, then  
    T(n) = Θ(nc) = Θ(f(n)).
2. If f(n) = Θ(nc), where logba = c, then  
    T(n) = Θ(nclog n) = Θ(f(n) log n).
3. If f(n) = Θ(nc), where logba > c, then  
    T(n) = Θ.

**Theorem 4.11:**

Let a and b be positive real numbers, with a ≥ 1 and b ≥ 2.Let T(n) satisfy the recurrence



Then we have the following:

1. If f(n) = Θ(nc), where logba < c, then  
    T(n) = Θ(nc) = Θ(f(n)).
2. If f(n) = Θ(nc), where logba = c, then  
    T(n) = Θ(nclog n) = Θ(f(n) log n).
3. If f(n) = Θ(nc), where logba > c, then  
    T(n) = Θ.

2.10 Giới thiệu công cụ lập trình symbolic: MAPLE

([tham khảo](file:///E:\\DayHoc\\Lap_trinh_tinhtoan%20&%20Computer%20Algebra))

* Cấu trúc kiểu dữ liệu và các tính toán cơ bản.
* Lập trình
* Thư viện các “package”.

Yêu cầu chung:

1. Xem xét một problem nào đó:  
   + Loại 1: problem ứng dụng cụ thể (hay lớp bài toán ứng dụng)  
   + Loại 2: Problem ở dạng lý thuyết (mô hình – căn cứ trên ứng thực tế)
2. Mục tiêu đồ án:

+ xây dựng thuật giải cho problem (nếu chưa có thuật giải)  
+ Trường hợp đã có thuật giải, thì mình tìm cách cải tiến thuật giải   
+ Căn cứ trên Problem đã được xét, đặt ra problem mới mở rộng hơn, nâng cao, hay đặc thù hơn. (có ý nghĩa)

1. Thực hiện và nộp đồ án:  
   + Báo cáo  
   + kèm theo …
2. **Bài toán tìm kiếm tài liệu (trên máy tính) theo một cụm từ dựa trên nội dung của tài liệu** 
   1. Căn cứ dữ liệu: tập các file TEXT   
      D = { d1, d2, …, dn }
   2. Input: cụm từ w
   3. Output: một tập D’ (nằm trong D) các tài liệu “liên quan” đến w
   4. Giới hạn:   
      \* các tập tin trong D là các tài liệu (sách, bài báo) về khoa học máy tính  
      \* nghĩa của “liên quan” giữa w với d: số lần xuất hiện của w trong d cao.

Giải pháp (thuật giải) cho vấn đề ?

**References:**

[1] Van Nhon Do, Thanh Thuong T. Huynh, Truong An Pham Nguyen. [Semantic Representation and Search Techniques for Document Retrieval Systems](file:///C:\Users\ConferenceJournalPublisher\2013,%20ACIIDS%202013\Final\ACIIDS2013,Part1,%20LNAI%2078020476.pdf)*.* In: ali Selamat, Ngoc Thanh Nguyen, Habibollah Haron (eds.) ACIIDS 2013, Part I, LNAI, vol. 7802, pp. 476-486. Springer-Verlag Berlin Heidelberg (2013).

1. **Các bài toán trên đồ thị lớn** 
   1. Đồ thị lớn và mẫu ví dụ (để chạy thử)
   2. Bài toán:   
      + Tìm đường đi ngắn nhất  
      + Người giao hàng (đưa thư)  
      + Minimal Spanning Tree  
      v.v…

* Bài toán tìm đường đi ngắn nhất trên đồ thị:

**Dạng 1a:**

Cho một đồ thị có trọng số dương: G = (V, E), ký hiệu trọng số là w. Với a, z V, tìm đường đi ngắn nhất từ a đến z.

**Base:**

G = (V, E) là đồ thị có trọng số dương.

Ước lượng khoảng cách từ mỗi đỉnh x đến đỉnh mục tiêu z, ký hiệu là h(x).

Định hướng ĐTNB.

Một số (không ít) đường đi ngắn nhất đã biết trên đồ thị.

**Input:**

Đỉnh xuất phát a

Đỉnh mục tiêu z

**Output:**

Đường đi ngắn nhất SP từ a đến z.

**Condition:**

0 <= h(x) <= h\*(x)

Đã có thuật giải A\*.

* Bài toán “Người giao hàng (đưa thư)”

**Base:**

G = (V, E) là đồ thị có trọng số dương.

**Input:**

**Output:**

Chu trình hamilton ngắn nhất.

* Bài toán “So khớp đồ thị”:

**Dạng 1: Bài toán “đẳng cấu đồ thị”**

**Base:**

**Input:**

Hai đồ thị G = (V,E) và G’=(V’,E’)

**Output:**

Yes/No: có/không sự đẳng cấu giữa G và G’.

Trường hợp “Yes”, hãy chỉ ra một phép đẳng cấu.

**Condition: Chỉ xét một số trường hợp**

1. Đồ thị vô hướng đơn giản.
2. Đồ thị có hướng đơn giản.

**Thuật giải vét cạn:**

Duyệt các song ánh f từ V vào V’, ứng với mỗi f ta kiểm tra ánh xạ f~ cho ứng với cạnh (a,b) của G là cạnh (f(a), f(b)) xem f~ có phải là song ánh không; nếu f~ là song ánh thì f là một phép đẳng cấu giữa G và G’.

**Dạng 2: Bài toán “So khớp đồ thị H trong đồ thị G”**

**Base:**

**Input:**

Hai đồ thị H = (VH,EH) và G = (V, E)

**Output:**

Yes/No: có/không sự đẳng cấu giữa H và một đồ thị con của G.

Trường hợp “Yes”, hãy chỉ ra một đồ thị con G1 và phép đẳng cấu giữa H và G1.

**Condition: Chỉ xét một số trường hợp**

1. Đồ thị vô hướng đơn giản.
2. Đồ thị có hướng đơn giản.

* Đồ thị khái niệm
* Đồ keypharse

**Chương 3:Xây dựng giải pháp cho vấn đề**

* 1. **Mô hình hóa vấn đề**
* **Các bước thực hiện:**Vấn đề thực tế 🡪 Vấn đề cần giải quyết 🡪 mô hình hóa vấn đề
* Vấn đề thực tế và vấn đề cần giải quyết:

+ Khảo sát và thu thập dữ liệu, thông tin và tri thức (DIK).

+ Chọn lọc vấn đề và chuẩn hóa DIK.

+ Xác định cơ sở DIK cho vấn đề.

+ Xác định phạm vi gay giới hạn của vấn đề.

+ Thu thập mẫu vấn đề và phân lớp.

+ Mô tả giả thiết của vấn đề.

+ Mô tả mục tiêu hay kết luận của vấn đề.

+ Mô tả các điều kiện hay ràng buộc liên quan.

* Xây dựng mô hình:

+ Mô hình cho DIK.

+ Mô hình cho giả thiết.

+ Mô hình cho mục tiêu.

+ Mô hình cho các điều kiện và các ràng buộc.

* Mô hình cho vấn đề tổng thể:

1. Dạng frame.
2. Dạng tổng quát.
3. Các dạng lai.

General Problem Solving

* Case study 1:
  1. Bài toán tìm đường đi ngắn nhất trên mạng giao thông.
  2. Bài toán 8-puzzle.
* Case study 2: Bài toán phân công.

2.1 Dạng đơn giản

2.2 Các dạng mở rộng

* Case study 3: Bài toán người đưa thư.
* Case study 4: Mô hình SDB và vấn đề tìm kiếm tài liệu theo ngữ nghĩa.
* Case study 5: Bài toán giải tam giác và mô hình “Mạng tính toán”
  1. **Thiết kế thuật toán / thuật giải**
* How to design algorithms?
  + - * See “[The algorithm design Manual](file:///C:\Users\noka\Dropbox\Reference\2008,%20The%20Algorithm%20Design%20Manual%20(Springer%20-%202nd%20edition).pdf)” - How to design algorithms
      * Slide: “[Problem Solving and Algorithm Design](file:///C:\Users\noka\Dropbox\Reference\4,%20PS%20and%20Algorithm%20Design.ppt)”

+ Look for familiar things

+ Divide and Conquer ([xem thêm](file:///C:\\Users\\noka\\Dropbox\\Reference\\ThuatToan-DoPhucTap))

+ Top-Down Design

+ Object-Oriented Design

* Phương pháp thiết kế “thuật giải heuristic”:
  + - * Khái niệm và tầm quan trọng

+ Thủ tục để giải quyết một vấn đề (bài toán)

+ Mô phỏng cách suy nghĩ giải quyết vấn đề của con người, đ/b là những vấn đề phức tạp 🡪 tính thông minh thể hiện ở những điểm sau đây:  
 (1) khai thác mọi thông tin hữu ích đối với vấn đề.  
 (2) yếu tố kinh nghiệm.  
 (3) yếu tố cảm tính, trực giác  
 (4) tự nhiên, đơn giản

* Cách giải tốt nhất trong tình huống hiện tại, để dẫn tới kết quả chấp nhận được (chưa chắc đúng như mong muốn)
  + - * Các nguyên lý heuristic:

+ Vét cạn thông minh

+ Tham lam

+ Thứ tự

+ Hàm heuristic

* + - * Một số thuật giải tham khảo: A\*, …

References:

[1] [slides](file:///C:\Users\noka\Dropbox\Reference\Heuristics%20and%20metaheuristic%20methods)

[2] Books: [1](file:///C:\Users\noka\E-Library\2016,%20Hybrid%20metaheuristics%20-%20Springer.pdf), [2](file:///C:\Users\noka\E-Library\2016,%20Workshop%20-%20Hybrid%20metaheuristics%20-%20Springer.pdf), [3](file:///C:\Users\noka\Dropbox\Reference\Heuristics%20and%20metaheuristic%20methods\Metaheuristics-%20Intelligent%20Problem%20Solving.pdf)

* Một số phương pháp giải quyết vấn đề thông dụng đã biết:
* **Phương pháp trực tiếp** dựa trên các dạng bài toán đã được nghiên cứu, các giải pháp có sẵn thường là các thuật toán, các công thức, các định lý, …  
  Ví dụ: các vấn đề và thuật toán tính toán số, các vấn đề và thuật toán đồ thị,
* **Phương pháp gián tiếp** chuyển bài toán về dạng thông dụng thường có những giải pháp đã được nghiên cứu. Chẳng hạn, bài toán tìm kiếm trên không gian trạng thái với các thuật giải quen thuộc như:…  
  + Phép duyệt DFS, BFS  
  + Phép duyệt theo mục tiêu tối ưu.  
  Các dạng bài toán đồ thị thông dụng đã biết giải pháp như bài toán đường đi, bài toán chu trình, bài toán cây bao trùm tối tiểu, …
* Phương pháp vét cạn, và vét cạn thông minh:  
  + Vét cạn:  
  + Vét cạn thông minh:
* Phương pháp quy lui

**Bài tập: Thiết kế và phân tích thuật toán**

1. **So khớp đồ thị**

* Phát biểu bài toán (ở dạng mô hình)
* Thiết kế thuật giải
* Đánh giá hiệu quả (tính độ phức tạp)

1. Tìm chu trình hamilton
2. Tìm đường đi trên bản đồ
3. Bài toán tìm luật từ một bảng CSDL

Dạng đơn giản:

X1, X2, …, Xm, Y

Có một bảng CSDL gồm n mẫu tin (n rất lớn),

Trong đó có m thuộc tính độc lập x1, … , xm và thuộc tính y là thuộc tính kết quả phụ thuộc theo các thuộc tính kia.

Tìm **quy luật** phụ thuộc của y theo x1, … , xm

Trường hợp 1: các biến đề là số thực

Trường hợp 2: các biến đều có kiểu rời rạc và hữu hạn

* Quy bài bài toán về dạng khác đơn giản hơn.
* Phân rã bài toán thành những bài toán đơn giản hơn
* Hình thành ý tưởng thiết kế thuật toán
* Mô tả cụ thể: tác vụ, chiến lượ:c, …
* Biểu diễn thuật toán dạng mã giả
  1. **Chứng minh tính đúng đắn**
* Chứng minh lý thuyết (dựa trên toán học)
* Kiểm chứng bằng thực nghiệm
  1. **Phân tích thuật toán / thuật giải**
* Sử dụng công cụ toán học
  1. **Nghiên cứu cải thiện, nâng cao hiệu quả thuật giải**
* Phương pháp “phân chia và kết hợp”
* Phương pháp heuristic
* Phương pháp giải quyết vấn đề dựa trên (dùng) tri thức

### Chương 4: Các phương pháp giải quyết vấn đề

**4.1Phương pháp trực tiếp:**

- áp dụng trực tiếp công thức hay định lý, algorithm

- chuyển đổi dạng bài toán.

- devide and conquer,

**4.2 Các phương pháp gián tiếp hoặc tìm kiếm lời giải thông dụng:**

- Phương pháp thử sai,

- Phương pháp vét cạn,

- Phương pháp quay lui,

- phương pháp đệ quy

Ví dụ 1: Bài toán 8 hậu

Ví dụ 2: Mã đi tuần

Ví dụ 3: Vượt mê cung

Ví dụ 4: Quick sort

4.3 **Các phương pháp heuristic:  
 Thiết kế “thuật giải heuristic” (heuristic algorithm)**

* Vét cạn thông minh
* Tham lam, và quy hoạch động
* Thứ tự
* Hàm heuristic

[Tham khảo 1](file:///C:\Users\noka\Dropbox\Reference\Heuristics%20and%20metaheuristic%20methods).

[Tham khảo 2](file:///E:\DayHoc\Algorithm-ProblemSolving\Dynamic%20Programming%20Slide.ppt).

**4.4. Phương pháp giải quyết vấn đề dùng tri thức (HCG)**

* Suy luận cơ bản
* suy luận với heuristic
* suy luận với mẫu bài toán
* suy luận với bài toán mẫu

**Tham khảo:**[**IPS**](file:///E:\DayHoc\IPS%20-%20Design%20method%20and%20Applications)**.**

4.5 case study và bài tập

**Loại đồ án:**

1. **Áp dụng**
2. **Tìm hiểu lý thuyết hay công cụ mới liên quan**
3. **Nghiên cứu cải tiến, nâng cao hiệu quả của thuật giải cho một hay một số lớp vấn đề**
4. **Mở rộng vấn đề đã có và xây dựng giải pháp.**
5. **Bài toán sắp thời khóa biểu**

* Phát biểu bài toán (dạng tự nhiên):
* DIK: CSDL về phòng học, lớp học, môn học, ngày học, buổi, ca, GV, …  
  + các thuộc tính phòng học: sức chứa, loại phòng, …  
  + môn học: tên môn, số chỉ, …  
  +
* Input:  
  + DS môn học  
  + DS GV – môn học  
  + DS lớp học
* Output:  
  + Bảng TKB: …
* Ràng buộc:  
  + Các ràng buộc cứng:  
   (RC1) ….  
   (RC2) ….  
  + Các ràng buộc mềm:  
   (RM1) …  
   (RM2) …
* Mô hình bài toán (dạng toán học hay đặc tả):
* Mô hình cho DIK:   
  P = {p1, p2, … } tập các phòng học, npi : sức chứa của phòng pi;   
  ….
* Mô hình input:  
  + MH = {mh1, mh2, … }, nMH : số môn học, …  
  + GV = { gv1, gv2, … }, nGV : số GV;  
  + GV\_MH = { các cặp (gv, mh) }   
  …
* Mô hình output:  
  + mô hình hóa “bảng TKB”  
   TKB\_GV\_MH = { các cặp (gv,mh) }  GV\_MH.
* Mô hình các ràng buộc:  
  +
* Một số pp tiếp cận:
* PP1: Thuật giải heuristic
* PP2: Thuật giải di truyền

1. **Bài toán mã hóa**

* Các thuật toán số học([xem](file:///C:\\Users\\DayHoc\\Phuong%20phap%20Toan%20cho%20Tin%20hoc\\coding%20theory\\Ma%20hoa%20thong%20tin.doc))
* Một phương pháp mã hóa

Vấn đề 3: Tìm lộ trình đi xe BUS giữa 2 trạm.

Vấn đề 4: Bài toán tìm luật từ một bảng CSDL

Dạng đơn giản:

X1, X2, …, Xm, Y

Có một bảng CSDL gồm n mẫu tin (n rất lớn),

Trong đó có m thuộc tính độc lập x1, … , xm và thuộc tính y là thuộc tính kết quả phụ thuộc theo các thuộc tính kia.

Tìm **quy luật** phụ thuộc của y theo x1, … , xm

Trường hợp 1: các biến đề là số thực

Trường hợp 2: các biến đều có kiểu rời rạc và hữu hạn

Vấn đề 5: Bài toán phân công công việc.

Vấn đề 6: Tìm đường đi trên bản đồ giao thông TP.HCM.

Vấn đề 7:

* So khớp đồ thị;
* Tính độ tương đồng ngữ nghĩa giữa 2 đồ thị keypharse hay giữa 2 đồ thị khái niệm

Vấn đề 8: Tối ưu hóa một số thuật toán đồ thị

* Tìm cây khung tối tiểu của đồ thị
* Tìm chu trình Hamilton

Vấn đề 9: Tối ưu hóa thuật giải giải vấn đề trên mạng tính toán.

Vấn đề 10: Đồ thị lớn, động, ngẫu nhiên

Vấn đề 11: Một số vấn đề trên “Big data”

Vấn đề 12: Thuật giải suy diễn giải quyết vấn đề trên CSTT

Dạng tổng quát của “vấn đề suy diễn”:

* Cơ sở tri thức: K
* Giả thiết: H
* Mục tiêu: G
* Điều kiện, ràng buộc: C

Xây dựng “**thuật giải suy diễn**”:

Input: H, G

Output: “lời giải”

Nghiên giải quyết cho những trường hợp với mô hình cụ thể hơn:

1. **Chọn mô hình K, và miền tri thức cụ thể để minh họa.**
2. **Mô hình cho H, G**
3. Thiết kế thuật giải
4. (option) thể hiện tính thông minh, tự nhiên, gần gũi với con người.
5. Tính đúng?
6. Hiệu quả? (nâng cao hiệu quả)
7. Thử nghiệm

**Trường hợp 1:**

K = (F, R)

F = { các biến logic }

R = { các luật dẫn }

r  R có dạng: h(r)  g(r), với h(r)  F và g(r)  F.

H  F, G  F

“lời giải” (solution): list L các r  R thỏa: L(H)G.

L(H) = tập sự kiện có được từ H do lần lượt áp dụng các luật trong danh sách L

Thuật giải suy diễn tiến có độ phức tạp là: O(|R|.|F|)

**Trường hợp 2:**

K = (M, F) mạng tính toán

M = { các biến số thực }

F = { các luật dạng phương trình }

H  M được cho trước giá trị, G  M cần tính ra.

“lời giải” (solution): list L các f  F thỏa: L(H)G.

L(H) = tập các biến tính được giá trị từ H do lần lượt áp dụng các phương trình trong danh sách L.

(Mở rộng L: danh sách các tập pt trong F thỏa L(H)G)

**Trường hợp 3:**

K = (O, F) mạng các đối tượng tính toán

O = { các đối tượng tính toán }

F = { các luật dạng phương trình trên các thuộc tính của các đối tượng }

Đặt M(obj) = tập các thuộc tính của đối tượng obj

M = hợp của tất cả các M(obj), với objO.

H  M được cho trước giá trị, G  M cần tính ra.

“lời giải” (solution): list L các f  F thỏa: L(H)G.

L(H) = tập các biến tính được giá trị từ H do lần lượt áp dụng các phương trình trong danh sách L.

(Mở rộng F = { các luật dạng phương trình trên các thuộc tính của các đối tượng hay trên các đối tượng } )

Vấn đề 13: Thuật giải di truyền (genetic algorithm) và ứng dụng

### Chương 5: NP-complete problems

5.1. Các lớp độ phức tạp và lớp P

5.2. Lớp NP

5.3. Lớp NP- đầy đủ

5.4. Ví dụ

### Chương 6: Thuật giải cho những vấn đề khó

6.1. Thuật giải xấp xĩ

6.2. Local Search

6.3. Một số thuật giải metaheuristic

6.4. Thuật giải di truyền và tiến hóa.

6.5. Mạng Neural.