Kỹ thuật tìm kiếm sâu lặp kết hợp được các ưu điểm của tìm kiếm theo bề rộng và tìm kiếm theo độ sâu. Chúng ta có một số nhận xét sau:

- Cũng như tìm kiếm theo bề rộng, tìm kiếm sâu lặp luôn luôn tìm ra nghiệm (nếu bài toán có nghiệm), miễn là ta chọn độ sâu max đủ lớn.
- Tìm kiếm sâu lặp chỉ cần không gian nhớ như tìm kiếm theo độ sâu.
- Trong tìm kiếm sâu lặp, ta phải phát triển lặp lại nhiều lần cùng một trạng thái. Điều đó làm cho ta có cảm giác rằng, tìm kiếm sâu lặp lãng phí nhiều thời gian. Thực ra thời gian tiêu tốn cho phát triển lặp lại các trạng thái là không đáng kể so với thời gian tìm kiếm theo bề rộng. Thật vậy, mỗi lần gọi thủ tục tìm kiếm sâu hạn chế tới mức d, nếu cây tìm kiếm có nhân tố nhánh là b, thì số đỉnh cần phát triển là:

$$1 + b + b^2 + \dots + b^d$$

Nếu nghiệm ở độ sâu d, thì trong tìm kiếm sâu lặp, ta phải gọi thủ tục tìm kiếm sâu hạn chế với độ sâu lần lượt là 0, 1, 2,..., d. Do đó các đỉnh ở mức 1 phải phát triển lặp d lần, các đỉnh ở mức 2 lặp d-1 lần,..., các đỉnh ở mức d lặp 1 lần. Như vậy tổng số đỉnh cần phát triển trong tìm kiếm sâu lặp là:

$$(d+1)1 + db + (d-1)b^2 + \dots + 2b^{d-1} + 1b^d$$

Do đó thời gian tìm kiếm sâu lặp là $O(b^d)$.

Tóm lại, tìm kiếm sâu lặp có độ phức tạp thời gian là $O(b^d)$ (như tìm kiếm theo bề rộng), và có độ phức tạp không gian là $O(bd)$ (như tìm kiếm theo độ sâu). Nói chung, chúng ta nên áp dụng tìm kiếm sâu lặp cho các vấn đề có không gian trạng thái lớn và độ sâu của nghiệm không biết trước.

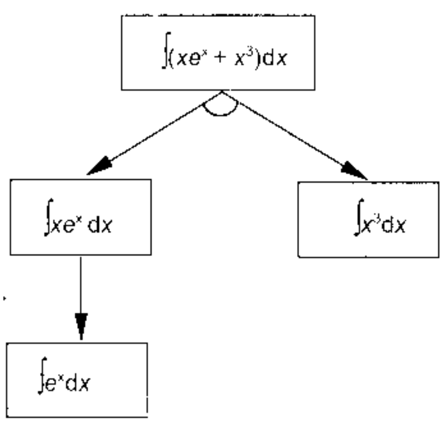
1.4. QUY VẤN ĐỀ VỀ CÁC VẤN ĐỀ CON. TÌM KIẾM TRÊN ĐỒ THỊ VÀ/HOẶC

1.4.1. Quy vấn đề về các vấn đề con

Trong mục 1.1, chúng ta đã nghiên cứu việc biểu diễn vấn đề thông qua các trạng thái và các toán tử. Khi đó việc tìm nghiệm của vấn đề được quy về việc tìm đường trong không gian trạng thái. Trong mục này chúng ta sẽ nghiên cứu một phương pháp luận khác để giải quyết vấn đề, dựa trên việc quy vấn đề về các vấn đề con. Quy vấn đề về các vấn đề con (còn gọi là rút gọn vấn đề) là một phương pháp được sử dụng rộng rãi nhất để giải quyết các vấn đề. Trong đời sống hàng ngày, cũng như trong khoa học kỹ thuật, mỗi khi gặp một vấn đề cần giải quyết, ta vẫn thường cố gắng tìm cách đưa nó về các vấn đề đơn giản hơn. Quá trình rút gọn vấn đề sẽ được tiếp tục cho tới khi ta dẫn tới các vấn đề con có thể giải quyết được dễ dàng. Sau đây chúng ta xét một số vấn đề.

VẤN ĐỀ TÍNH TÍCH PHÂN BẤT ĐỊNH

Giả sử ta cần tính một tích phân bất định, chẳng hạn $\int (xe^x + x^3)\,dx$. Quá trình chúng ta vẫn thường làm để tính tích phân bất định là như sau. Sử dụng các quy tắc tính tích phân (quy tắc tính tích phân của một tổng, quy tắc tính tích phân từng phần...), sử dụng các phép biến đổi biến số, các phép biến đổi các hàm (chẳng hạn, các phép biến đổi lượng giác),... để đưa tích phân cần tính về tích phân của các hàm số sơ cấp mà chúng ta đã biết cách tính. Đối với tích phân $\int (xe^x + x^3)dx$, áp dụng quy tắc tích phân của tổng ta đưa về hai tích phân $\int xe^x dx$ và $\int x^3 dx$. Áp dụng quy tắc tích phân từng phần ta đưa tích phân $\int xe^x dx$ về tích phân $\int e^x dx$. Quá trình trên có thể biểu diễn bởi đồ thị trong hình 1.5.



***Hình 1.5.* Quy một tích phân về các tích phân cơ bản.**

Các tích phân $\int e^x dx$ và $\int x^3 dx$ là các tích phân cơ bản đã có trong bảng tích phân. Kết hợp các kết quả của các tích phân cơ bản, ta nhận được kết quả của tích phân đã cho.

Chúng ta có thể biểu diễn việc quy một vấn đề về các vấn đề con bởi các trạng thái và các toán tử. Ở đây, bài toán cần giải là trạng thái ban đầu. Mỗi cách quy bài toán về các bài toán con được biểu diễn bởi một toán tử, toán tử $A \to B, C$ biểu diễn việc quy bài toán A về hai bài toán B và C. Chẳng hạn, đối với bài toán tính tích phân bất định, ta có thể xác định các toán tử dạng:

$$\int(f_1 + f_2)dx \to \int f_1 dx, \int f_2 dx$$

$$\int u dv \to \int v du$$

Các trạng thái kết thúc là các bài toán sơ cấp (các bài toán đã biết cách giải). Chẳng hạn, trong bài toán tính tích phân, các tích phân cơ bản là các trạng thái kết thúc. Một điều cần lưu ý là, trong không gian trạng thái biểu diễn việc quy vấn đề về các vấn đề con, các toán tử có thể là đa trị, nó biến đổi một trạng thái thành nhiều trạng thái khác.

VẤN ĐỀ TÌM ĐƯỜNG ĐI TRÊN BẢN ĐỒ GIAO THÔNG

Bài toán này đã được phát biểu như bài toán tìm đường đi trong không gian trạng thái (xem 1.1), trong đó mỗi trạng thái ứng với một thành phố, mỗi toán tử ứng với một con đường, nối thành phố này với thành phố khác. Bây giờ ta đưa ra một cách biểu diễn khác dựa trên việc quy vấn đề về các vấn đề con. Giả sử ta có bản đồ giao thông trong một vùng lãnh thổ (xem hình 1.6). Giả sử ta cần tìm đường đi từ thành phố A tới thành phố B. Có con sông chảy qua hai thành phố E và G và có cầu qua sông ở mỗi thành phố đó. Mọi đường đi từ A đến B chỉ có thể qua E hoặc G. Như vậy bài toán tìm đường đi từ A đến B được quy về:

1) Bài toán tìm đường đi từ A đến B qua E (hoặc)

2) Bài toán tìm đường đi từ A đến B qua G.

Mỗi một trong hai bài toán trên lại có thể phân nhỏ như sau

1) Bài toán tìm đường đi từ A đến B qua E được quy về:

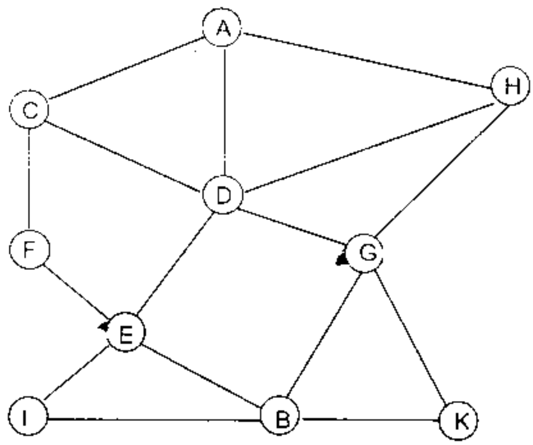
   1.1 Tìm đường đi từ A đến E (và)

   1.2 Tìm đường đi từ E đến B.

2) Bài toán tìm đường đi từ A đến B qua G được quy về:

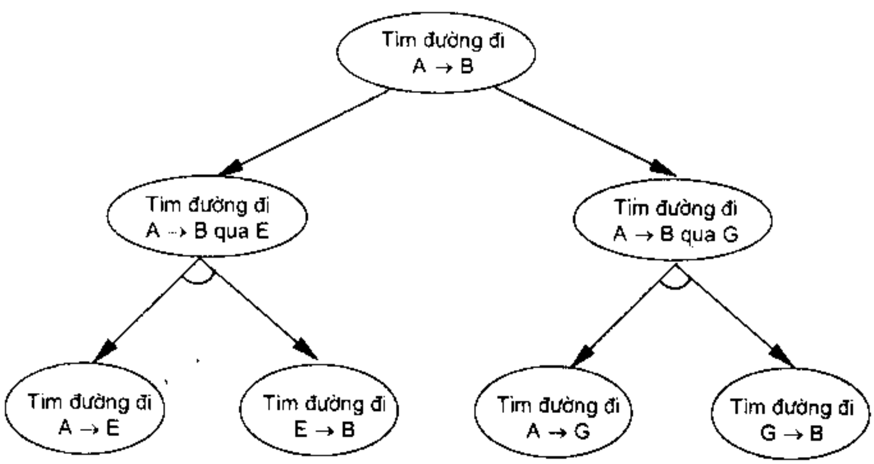
   2.1 Tìm đường đi từ A đến G (và)

   2.2 Tìm đường đi từ G đến B.



Hình 1.6. **Một mạng lưới giao thông.**

Quá trình rút gọn vấn đề như trên có thể biểu diễn dưới dạng đồ thị (đồ thị và/hoặc) trong hình 1.7. Ở đây mỗi bài toán tìm đường đi từ một thành phố tới một thành phố khác ứng với một trạng thái. Các trạng thái kết thúc là các trạng thái ứng với các bài toán tìm đường đi, chẳng hạn từ A đến C, hoặc từ D đến E, bởi vì đã có đường nối A với C, nối D với E.

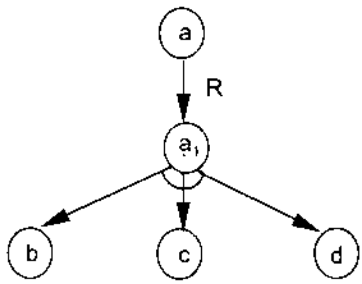


Hình 1.7. **Đồ thị và/hoặc của vấn đề tìm đường đi.**

1.4.2. Đồ thị và/hoặc

Không gian trạng thái mô tả việc quy vấn đề về các vấn đề con có thể biểu diễn dưới dạng đồ thị định hướng đặc biệt được gọi là đồ thị và/hoặc. Đồ thị này được xây dựng như sau:

Mỗi bài toán ứng với một đỉnh của đồ thị. Nếu có một toán tử quy một bài toán về một bài toán khác, chẳng hạn $R: a \rightarrow b$, thì trong đồ thị sẽ có cung gán nhãn R đi từ đỉnh a tới đỉnh b. Đối với mỗi toán tử quy một bài toán về một số bài toán con, chẳng hạn $R: a \rightarrow b, c, d$ ta đưa vào một đỉnh mới a_1, đỉnh này biểu diễn tập các bài toán con $\{b, c, d\}$ và toán tử $R: a \rightarrow b, c, d$ được biểu diễn bởi đồ thị hình 1.8.



Hình 1.8. **Đồ thị biểu diễn toán tử R: a → b, c, d.**

Ví dụ. Giả sử chúng ta có không gian trạng thái sau:

- Trạng thái ban đầu (bài toán cần giải) là a.
- Tập các toán tử quy gồm:

$R_1: a \rightarrow d, e, f$

$R_2: a \rightarrow d, k$

$R_3: a \rightarrow g, h$

$R_4: d \rightarrow b, c$

$R_5: f \rightarrow i$

$R_6: f \rightarrow c, j$

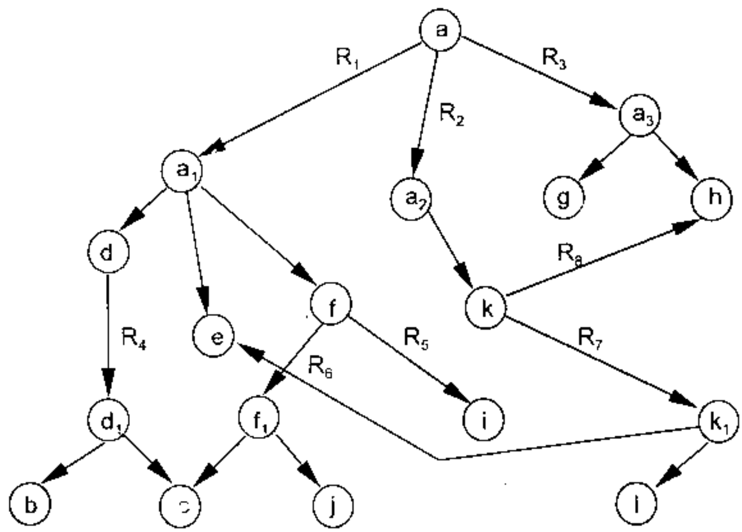
$R_7: k \rightarrow e, l$

$R_8: k \rightarrow h$

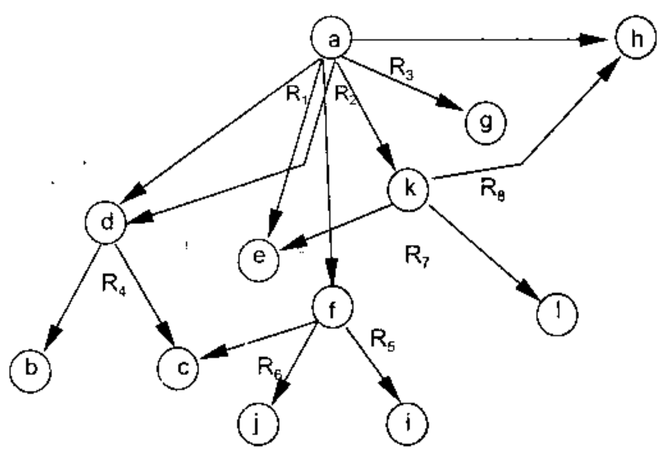
- Tập các trạng thái kết thúc (các bài toán sơ cấp) là $T = \{b, c, e, j, l\}$.

Không gian trạng thái trên có thể biểu diễn bởi đồ thị và/hoặc trong hình 1.9. Trong đồ thị đó, các đỉnh, chẳng hạn a_1, a_2, a_3 được gọi là đỉnh ***và***, các đỉnh chẳng hạn a, f, k được gọi là đỉnh ***hoặc***. Lý do là, đỉnh a_1 biểu diễn tập các bài toán $\{d, e, f\}$ và a_1 được giải quyết nếu d và e và f được giải quyết. Còn tại đỉnh a, ta có các toán tử R_1, R_2, R_3 quy bài toán a về các bài toán con khác nhau, do đó a được giải quyết nếu hoặc $a_1 = \{d, e, f\}$, hoặc $a_2 = \{d, k\}$, hoặc $a_3 = \{g, h\}$ được giải quyết.

Người ta thường sử dụng đồ thị và/hoặc ở dạng rút gọn. Chẳng hạn, đồ thị và/hoặc trong hình 1.9 có thể rút gọn thành đồ thị trong hình 1.10. Trong đồ thị rút gọn này, ta sẽ nói chẳng hạn d, e, f là các đỉnh kề đỉnh a theo toán tử R_1, còn d, k là các đỉnh kề a theo toán tử R_2.

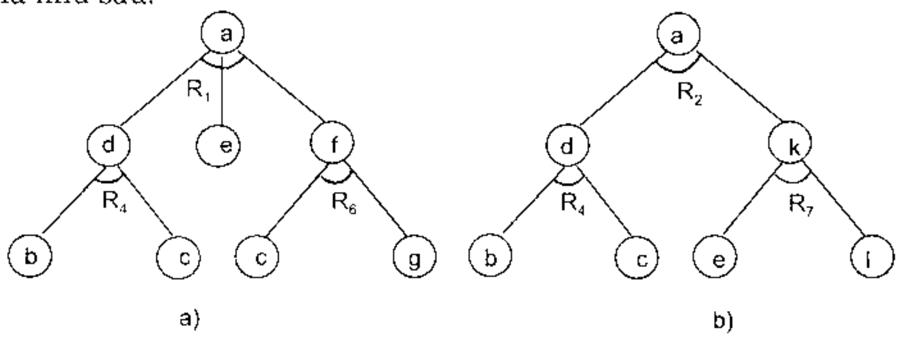


Hình 1.9. **Một đồ thị và/hoặc.**



Hình 1.10. **Đồ thị rút gọn của đồ thị trong hình 1.9.**

Khi đã có các toán tử rút gọn vấn đề, thì bằng cách áp dụng liên tiếp các toán tử, ta có thể đưa bài toán cần giải về một tập các bài toán con. Chẳng hạn, trong ví dụ trên, nếu ta áp dụng các toán tử R_1, R_4, R_6, ta sẽ quy bài toán a về tập các bài toán con $\{b, c, e, f\}$, tất cả các bài toán con này đều là sơ cấp. Từ các toán tử R_1, R_4 và R_6 ta xây dựng được một cây trong hình 1.11a, cây này được gọi là cây nghiệm. Cây nghiệm được định nghĩa như sau:



***Hình 1.11.* Các cây nghiệm.**

Cây nghiệm là một cây, trong đó:

- Gốc của cây ứng với bài toán cần giải.
- Tất cả các lá của cây là các đỉnh kết thúc (đỉnh ứng với các bài toán sơ cấp).
- Nếu u là đỉnh trong của cây, thì các đỉnh con của u là tất cả các đỉnh kề u theo một toán tử nào đó.

Các đỉnh của đồ thị và/hoặc sẽ được gắn nhãn giải được hoặc không giải được.

Các đỉnh ***giải được*** được xác định đệ quy như sau:

- Các đỉnh kết thúc là các đỉnh giải được.
- Nếu u không phải là đỉnh kết thúc, nhưng có một toán tử R sao cho tất cả các đỉnh kề u theo R đều giải được thì u giải được.

Các đỉnh ***không giải được*** được xác định đệ quy như sau:

- Các đỉnh không phải là đỉnh kết thúc và không có đỉnh kề, là các đỉnh không giải được.
- Nếu u không phải là đỉnh kết thúc và với mọi toán tử R áp dụng được tại u đều có một đỉnh v kề u theo R không giải được, thì u không giải được.

Ta có nhận xét rằng, nếu bài toán a giải được thì sẽ có một cây nghiệm gốc a, và ngược lại nếu có một cây nghiệm gốc a thì a giải được. Hiển nhiên là, một bài toán giải được có thể có nhiều cây nghiệm, mỗi cây nghiệm biểu diễn một cách giải bài toán đó. Chẳng hạn trong ví dụ đã nêu, bài toán a có hai cây nghiệm trong hình 1.11.

Thứ tự giải các bài toán con trong một cây nghiệm là như sau. Bài toán ứng với đỉnh u chỉ được giải sau khi tất cả các bài toán ứng với các đỉnh con của u đã được giải. Chẳng hạn, với cây nghiệm trong hình 1.11a, thứ tự giải các bài toán có thể là b, c, d, j, f, e, a. Ta có thể sử dụng thủ tục sắp xếp topo (xem [8]) để sắp xếp thứ tự các bài toán trong một cây nghiệm. Đương nhiên ta cũng có thể giải quyết đồng thời các bài toán con ở cùng một mức trong cây nghiệm.

Vấn đề của chúng ta bây giờ là, tìm kiếm trên đồ thị và/hoặc để xác định được đỉnh ứng với bài toán ban đầu là giải được hay không giải được, và nếu nó giải được thì xây dựng một cây nghiệm cho nó.

1.4.3. Tìm kiếm trên đồ thị và/hoặc

Ta sẽ sử dụng kỹ thuật tìm kiếm theo độ sâu trên đồ thị và/hoặc để đánh dấu các đỉnh. Các đỉnh sẽ được đánh dấu giải được hoặc không giải được theo định nghĩa đệ quy về đỉnh giải được và không giải được. Xuất phát từ đỉnh ứng với bài toán ban đầu, đi xuống theo độ sâu, nếu gặp đỉnh u là đỉnh kết thúc thì nó được đánh dấu giải được. Nếu gặp đỉnh u không phải là đỉnh kết thúc và từ u không đi tiếp được, thì u được đánh dấu không giải được. Khi đi tới đỉnh u, thì từ u ta lần lượt đi xuống các đỉnh v kề u theo một toán tử R nào đó. Nếu đánh dấu được một đỉnh v không giải được thì không cần đi tiếp xuống các đỉnh v còn lại. Tiếp tục đi xuống các đỉnh kề u theo một toán tử khác. Nếu tất cả các đỉnh kề u theo một toán tử nào đó được đánh dấu giải được thì u sẽ được đánh dấu giải được và quay lên cha của u. Còn nếu từ u đi xuống các đỉnh kề nó theo mọi toán tử đều gặp các đỉnh kề được đánh dấu không giải được, thì u được đánh dấu không giải được và quay lên cha của u.

Ta sẽ biểu diễn thủ tục tìm kiếm theo độ sâu và đánh dấu các đỉnh đã trình bày trên bởi hàm đệ quy Solvable(u). Hàm này nhận giá trị ***true*** nếu u giải được và nhận giá trị ***false*** nếu u không giải được. Trong hàm Solvable(u), ta sẽ sử dụng:

Biến *Ok*. Với mỗi toán tử *R* áp dụng được tại *u*, biến *Ok* nhận giá trị ***true*** nếu tất cả các đỉnh *v* kề *u* theo *R* đều giải được. và *Ok* nhận giá trị ***false*** nếu có một đỉnh *v* kề *u* theo *R* không giải được.

Hàm Operator(*u*) ghi lại toán tử áp dụng thành công tại *u*. tức là Operator(*u*) = *R* nếu mọi đỉnh *v* kề *u* theo *R* đều giải được.

```
Function   Solvable(u);
begin
   1. if  u là đỉnh kết thúc  then
          { Solvable ← true; stop };
   2. if  u không là đỉnh kết thúc và không có đỉnh kề  then
          { Solvable(u) ← false; stop };
   3. for  mỗi toán tử R áp dụng được tại u  do
          { Ok ← true;
          for  mỗi v kề u theo R  do
                 if  Solvable(v) = false  then
                        { Ok ← false; exit };
          if  Ok  then
                 { Solvable(u)← true;
                 Operator(u)← R; stop }
          }
   4. Solvable(u)← false;
end;
```

Nhận xét:

Hoàn toàn tương tự như thuật toán tìm kiếm theo độ sâu trong không gian trạng thái (mục 1.3.2), thuật toán tìm kiếm theo độ sâu trên đồ thị và/hoặc sẽ xác định được bài toán ban đầu là giải được hay không giải được. nếu cây tìm kiếm không có nhánh vô hạn. Nếu cây tìm kiếm có nhánh vô hạn thì chưa chắc thuật toán đã dừng. vì có thể nó bị xa lầy khi đi xuống nhánh vô hạn. Trong trường hợp này ta nên sử dụng thuật toán tìm kiếm sâu lặp (mục 1.3.3).

Nếu bài toán ban đầu giải được. thì bằng cách sử dụng hàm Operator ta sẽ xây dựng được cây nghiệm.

CHƯƠNG 2

CÁC CHIẾN LƯỢC TÌM KIẾM KINH NGHIỆM

Trong chương 1, chúng ta đã nghiên cứu việc biểu diễn vấn đề trong không gian trạng thái và các kỹ thuật tìm kiếm mù. Các kỹ thuật tìm kiếm mù rất kém hiệu quả và trong nhiều trường hợp không thể áp dụng được. Trong chương này, chúng ta sẽ nghiên cứu các phương pháp tìm kiếm kinh nghiệm (tìm kiếm heuristic), đó là các phương pháp sử dụng hàm đánh giá để hướng dẫn sự tìm kiếm.

2.1. HÀM ĐÁNH GIÁ VÀ TÌM KIẾM KINH NGHIỆM

Trong nhiều vấn đề, ta có thể sử dụng kinh nghiệm, tri thức của chúng ta về vấn đề để đánh giá các trạng thái của vấn đề. Với mỗi trạng thái u, chúng ta sẽ xác định một giá trị số $h(u)$, số này là số đo sự đánh giá của chúng ta về trạng thái u. Hàm $h(u)$ được gọi là ***hàm đánh giá***. Chúng ta sẽ sử dụng hàm đánh giá để hướng dẫn sự tìm kiếm. Trong quá trình tìm kiếm, tại mỗi bước ta sẽ chọn trạng thái để phát triển là trạng thái có giá trị hàm đánh giá nhỏ nhất, trạng thái này được xem là trạng thái có nhiều hứa hẹn nhất dẫn tới đích cần tìm kiếm.

Các kỹ thuật tịm kiếm sử dụng hàm đánh giá để hướng dẫn sự tìm kiếm được gọi chung là các kỹ thuật ***tìm kiếm kinh nghiệm*** (heuristic search). Các giai đoạn cơ bản để giải quyết vấn đề bằng tìm kiếm kinh nghiệm như sau:

1. Tìm biểu diễn thích hợp mô tả các trạng thái và các toán tử của vấn đề.
2. Xây dựng hàm đánh giá.
3. Thiết kế chiến lược chọn trạng thái để phát triển ở mỗi bước.