**CONSTRAINT SATISFACTION PROBLEM**

**(CSPs )**

Định nghĩa :

State : được xác định bởi :

Một tập hợp các biến X

**TÌM KIẾM BẰNG CÁCH KIỂM THỬ**

Ý tưởng:

* Sinh ra một khả năng ( candicate ) của lời giải
* Kiểm tra xem khả năng này có thực sự là 1 lời giải

Các bước

1. Gán các giá trị cho tất cả các biến
2. Kiểm tra xem tất cả các ràng buộc được thoả mãn hay không ?
3. Lặp lại 2 bước này cho đến khi tìm được một phép gán thoả mãn.

Ví dụ :

Các biến : A, B, C

Các giá trị : ( 1, 3 )

Các ràng buộc “ A = B, A khác C, B > C

Các phép gán ( có thể có ) :

~~( 1, 1, 1 ); ( 1, 1, 3 ); ( 1,3,1 ); (1,3,4); (3,1,1); (3,1,3); (1,3,3); (3,1,1); ( 3,1,1);~~ (3,3,1 )

Lọc dần các đối thù và cuối cùng ( 3, 3, 1 ) thoả.

Hạn chế :

* Phải xét quá nhiều các khả năng gán ( hiển nhiên ) không thoả mãn các ràng buộc.
* Tìm kiếm bằng kiểm thử : mỗi lần gán xác định các giá trị cho tất cả các biến.

**TÌM KIẾM QUAY LUI ( Back tracking )**

Ý tưởng :

* Dựa trên giải thuật tìm keiems theo chiều sâu ( depth first search )
* Mỗi lần gán, chỉ làm việc ( gán giá trị ) cho một biến

Phương pháp :

* Gán giá trị lần lượt cho các biến – việc gán giá trị của biến này chỉ được làm sau khi đã hoàn thành việc gán giá trị của biến khác
* Sau mỗi phép gnas giá tị cho một biến nào đó, kiểm tra các ràng buộc có được thoả mãn bởi tất cả các biến đã được gán giá trị cho đến thời điểm hiện tại. Quay lui ( backtrack ) nếu có lỗi.

Ví dụ : Backtracking search ( DFS )

Thực chất là thuật toán DFS nhưng chỉ phát triển cho 1 nhánh

**Chọn biến**

Quy tắc chọn thứ tự xét biến : ưu tiên biến bị ràng buộc nhiều nhất

Chọn biến có số lượng giá trị hợp lệ ít nhất

Ví dụ : Tại bước S2, biến NT được chọn vì nó có số lượng các giá trị hợp lệ ít nhất

Khi có >=2 biến có như nhau số lượng giá trị hợp lệ ít nhất thì chọn biến nào?

Ví dụ : trong ví dụ trước, 2 biến NT và SA có cùng số lượng giá trị hợp lệ ít nhất

Chọn biến ràng buộc ( khống chế ) các biến khác ( chưa được gán giá trị ) nhiều nhất.

Ví dụ : tại bước S2, tuy cùng mức độ bị ràng buộc nhưng biến SA nên được xét trước biến NT – vì SA ràng buộc 5 biến khác. Còn NT chỉ ràng buộc 3 biến khác.

**Kiểm tra tiến ( Forward Checking )**

Kiểm tra sớm các ràng buộc.

**Thuật toán tìm kiếm AC – 3**

**Tìm kiếm Min Conflict ( tìm hiểu )**