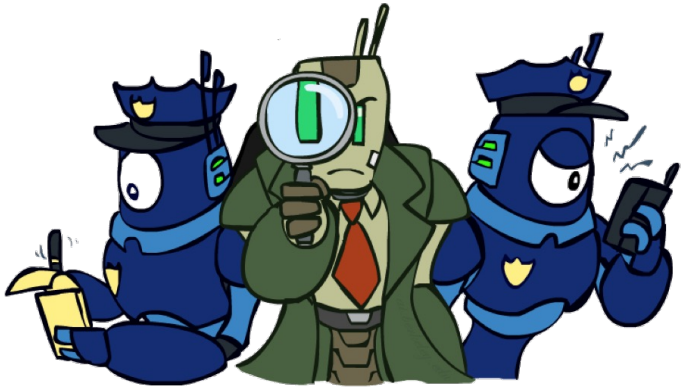
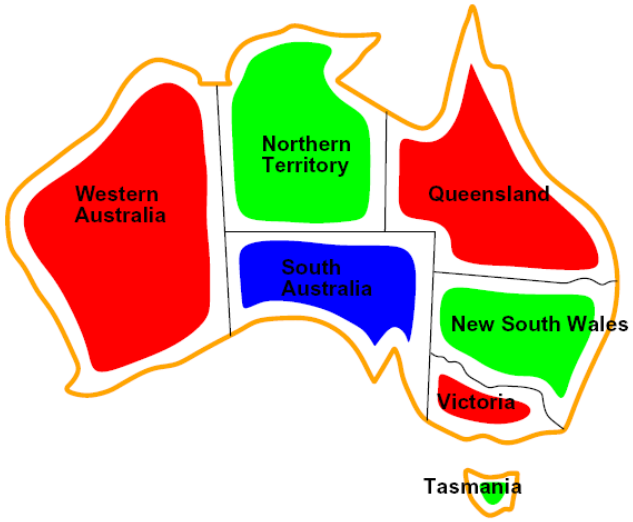
مسائل ارضای محدودیت

Constraint Satisfaction Problems (CSP)





بهبود جستجوی عمقی

۱. مقداردهی یک متغیر در هر سطح

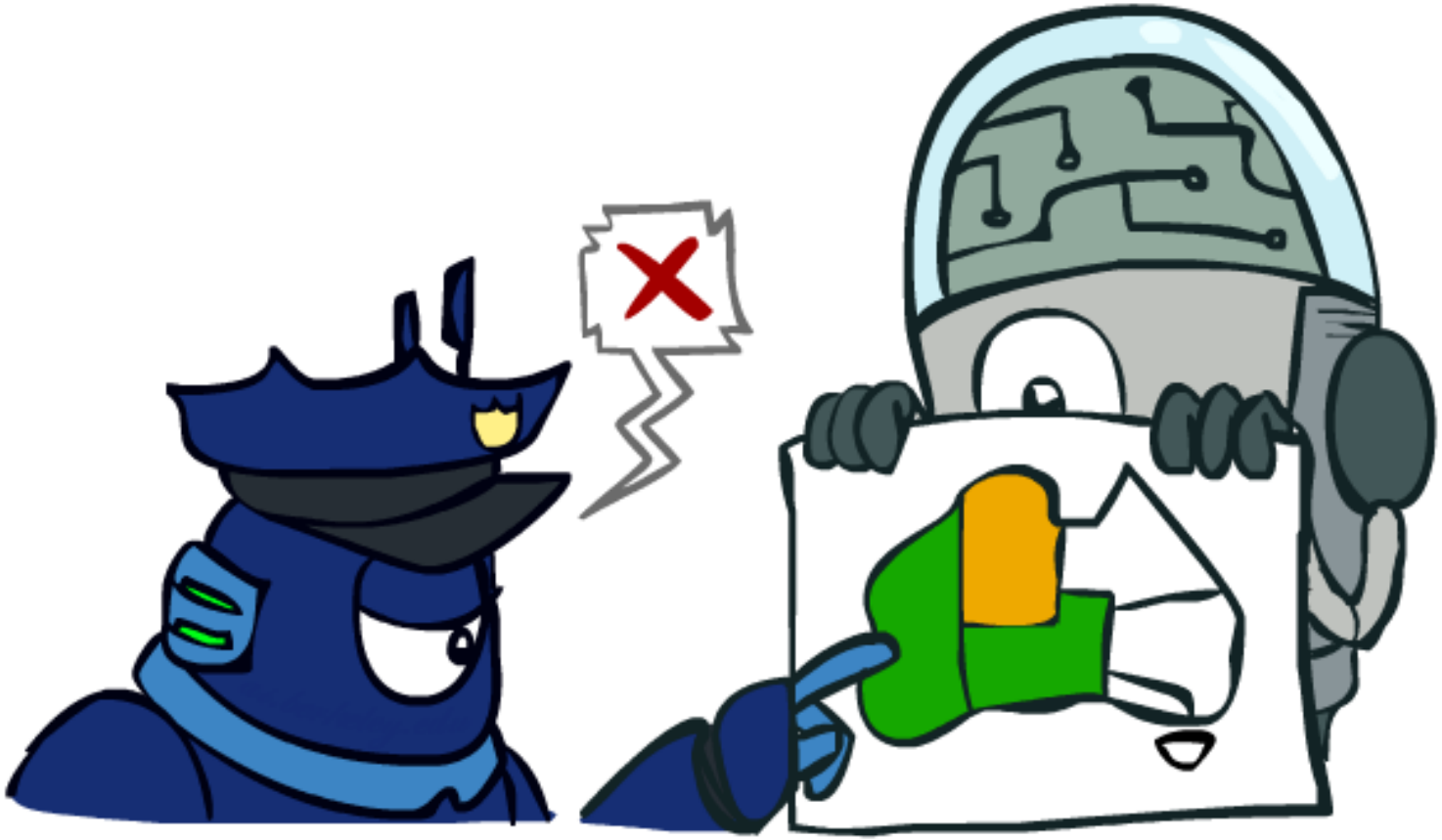
- در هرسطر به یک متغیر مقدار داده شود ، نه به همه متغیرها.
- بسیاری از مسیرها معادل یکدیگرند.

۲. حذف مسیرهای نامعتبر (هرس شاخه‌های نامعتبر)

- اگر در یک گره، محدودیتی نقض شود، دیگر زیرشاخه‌های آن را بررسی نمی‌کنیم، چون می‌دانیم که آن مسیر به جواب معتبر نخواهد رسید.

❑جستجوی عمقی همه مسیرها را تا انتها دنبال می‌کند، حتی مسیرهای نامعتبر، اما **با این دو بهینه‌سازی، الگوریتم به جستجوی عقب‌گرد تبدیل می‌شود.**

جستجوی عقب‌گرد (Backtracking Search)



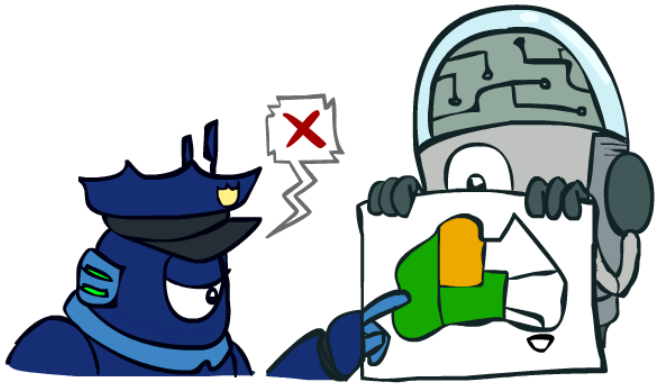
جستجوی عقب‌گرد

- جستجوی عقب‌گرد، یک الگوریتم پایه ناآگاهانه برای حل مسائل CSP

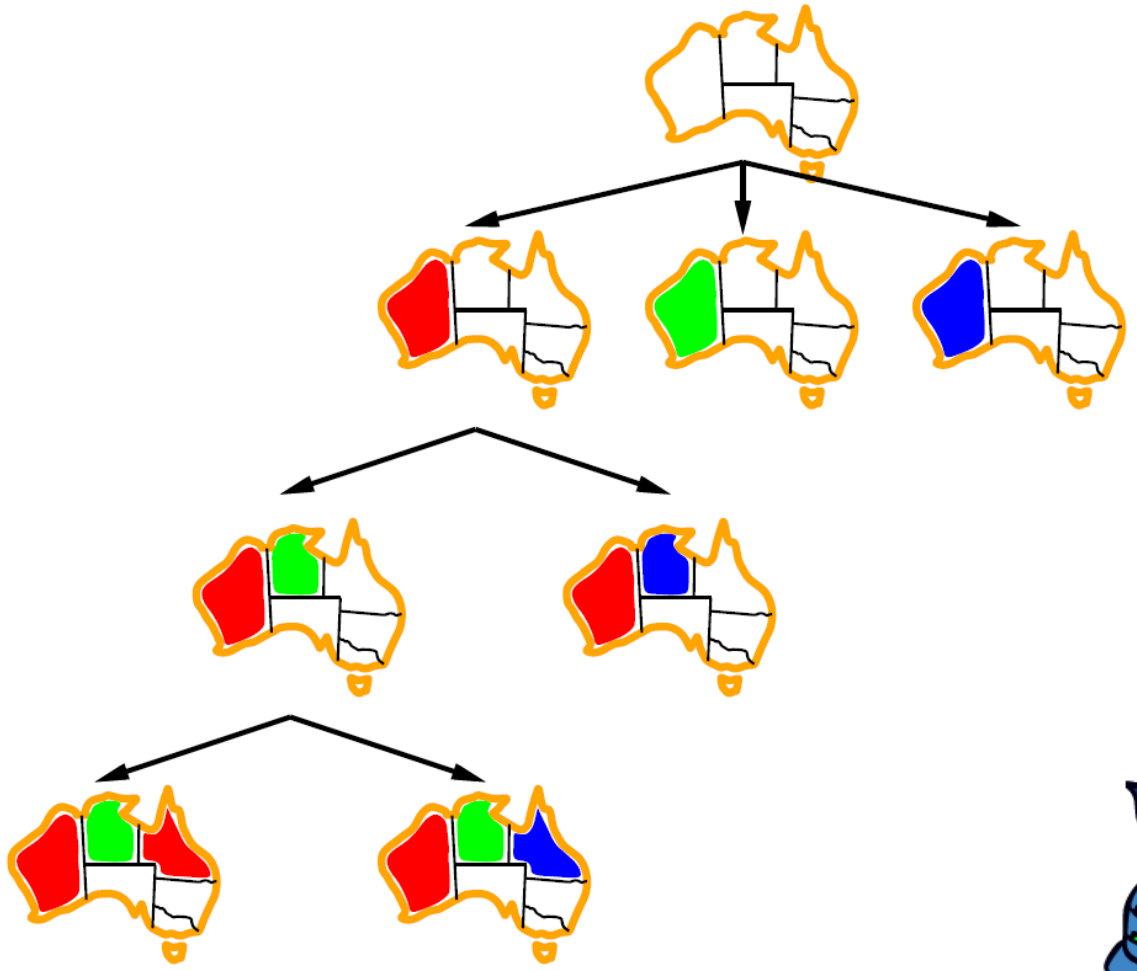
- **ایده 1: یک متغیر در هر مرحله.**
  - انتساب متغیرها جابجاپذیر (commutative) است، بنابراین یک ترتیب ثابت برای انتساب‌ها در نظر بگیر.
    - به‌عنوان مثال، [WA=قرمز سپس NT=سبز] معادل [NT=سبز سپس WA=قرمز] است.
    - در هر مرحله فقط کافی است مقداردهی را برای یک متغیر در نظر بگیریم.

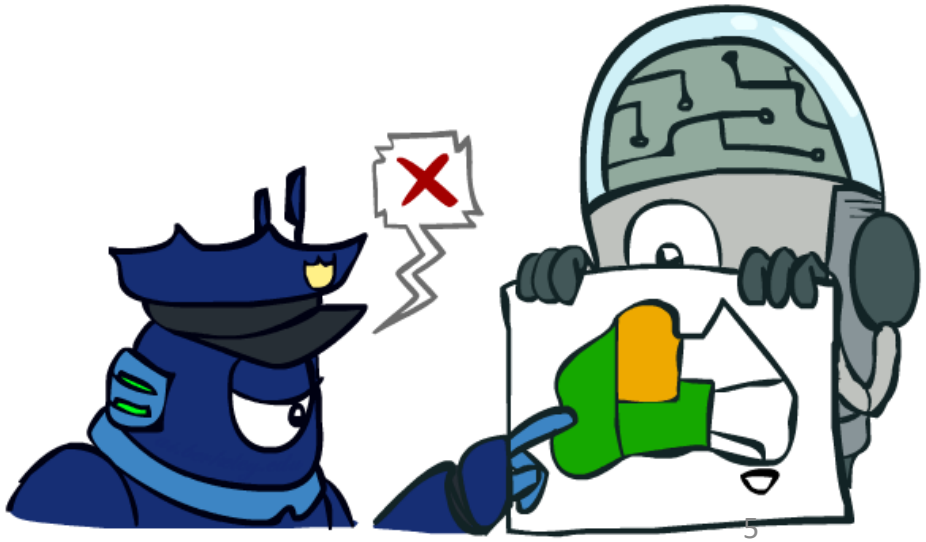
- **ایده 2: بررسی محدودیت‌ها در حین پیشروی.**
  - فقط مقادیری را در نظر بگیر که با انتساب‌های قبلی در تضاد نباشند.
  - ممکن است برای بررسی محدودیت‌ها نیاز به محاسبات داشته باشیم.
  - این فرآیند همان "آزمون هدف به صورت افزایشی" (Incremental Goal Test) است.

- جستجوی عمقی با این دو بهبود به جستجوی عقب‌گرد معروف است



مثال: جستجوی عقب‌گرد





ویدئوی نمایش رنگ‌آمیزی - جستجوی عقب‌گرد

جستجوی عقب‌گرد

```
function BACKTRACKING-SEARCH(csp) returns solution/failure
    return RECURSIVE-BACKTRACKING({ }, csp)

function RECURSIVE-BACKTRACKING(assignment, csp) returns soln/failure
    if assignment is complete then return assignment
    var ← SELECT-UNASSIGNED-VARIABLE(VARIABLES[csp], assignment, csp)
    for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
        if value is consistent with assignment given CONSTRAINTS[csp] then
            add {var = value} to assignment
            result ← RECURSIVE-BACKTRACKING(assignment, csp)
            if result ≠ failure then return result
            remove {var = value} from assignment
    return failure
```

ایده 1: یک متغیر در هر مرحله

ایده 2: بررسی محدودیت‌ها در حین پیشروی.

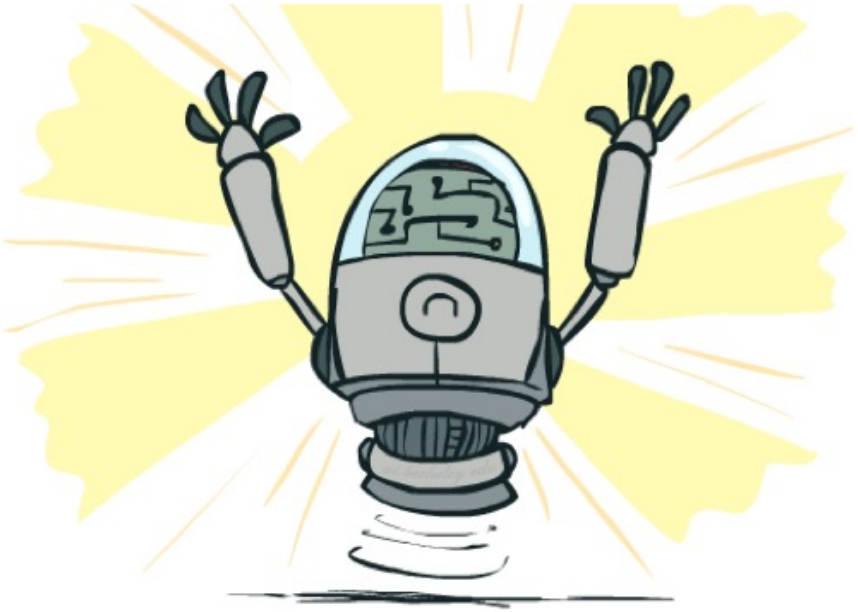
- جستجوی عقب‌گرد (Backtracking) = جستجوی عمقی (DFS) + یک متغیر در هر مرحله + بررسی محدودیت‌ها در حین پیشروی

بهبود جستجوی عقب‌گرد

- ایده‌های عمومی (همه-منظوره) باعث افزایش چشمگیر سرعت می‌شوند

- **ترتیب‌دهی (Ordering):**
  - کدام متغیر باید بعدی مقدار بگیرد؟
  - مقدارهای آن متغیر به چه ترتیبی باید امتحان شوند؟

- **فیلتر کردن (Filtering):**
  - آیا می‌توانیم شکست اجتناب‌ناپذیر را زودتر تشخیص دهیم؟

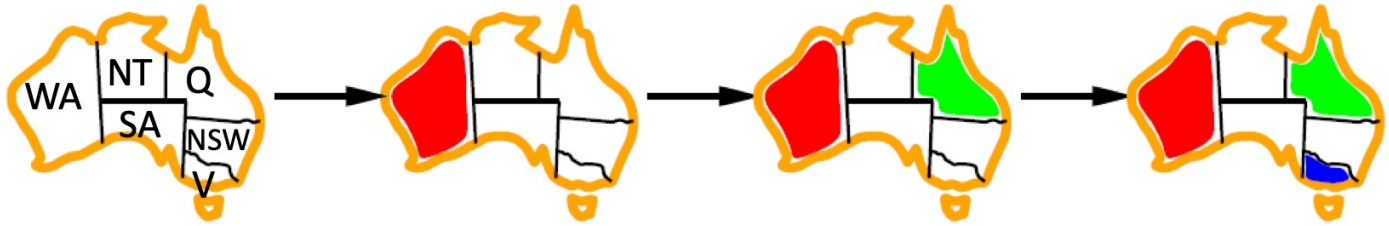


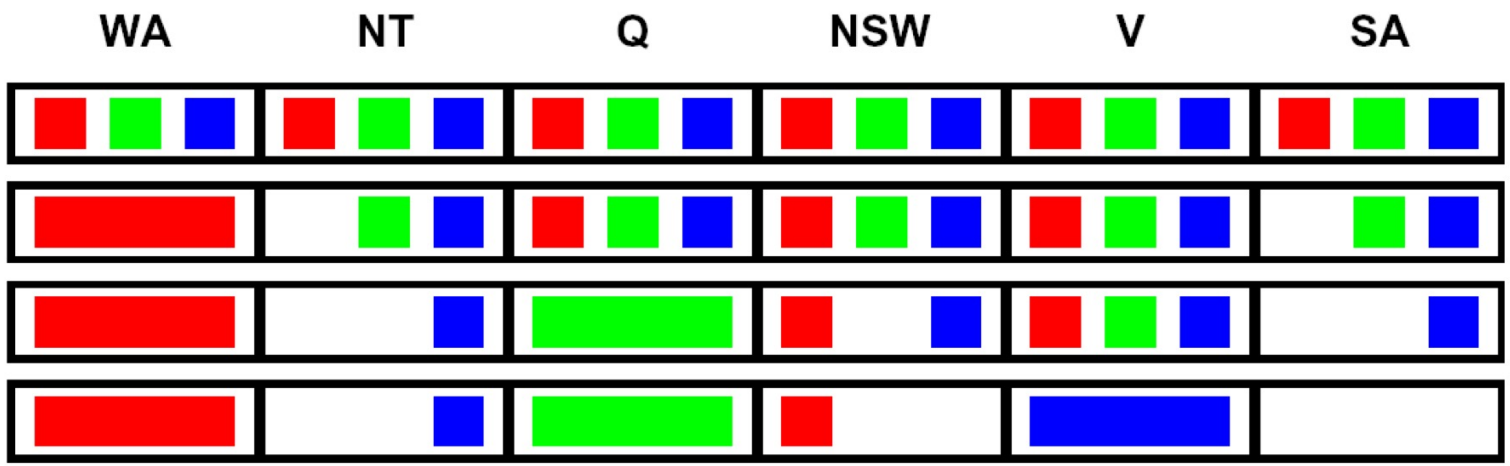
فیلتر کردن

دامنه‌های متغیرهای انتساب نیافته را بررسی کرده و گزینه‌های نامعتبر را حذف می‌کنیم.

فیلتر کردن: بررسی رو به جلو (Forward Checking)

- **فیلترکردن:** دامنه‌های متغیرهای انتساب نیافته را بررسی کرده و گزینه‌های نامعتبر را حذف می‌کنیم.
- **بررسی روبه جلو :** روشی است که هنگام مقداردهی یک متغیر، تأثیر این مقدار را روی متغیرهای دیگر بررسی می‌کند و مقادیر ناسازگار را از دامنه آن‌ها حذف می‌کند
  - اگر دامنه‌ی هر متغیر خالی شود، به این معنی است که انتخاب انجام‌شده منجر به بن‌بست شده و نیاز به **بازگشت (backtracking)** داریم.

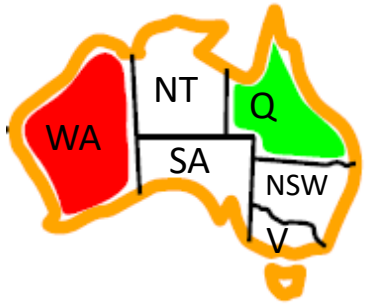




ویدئوی نمایش رنگ‌آمیزی - جستجوی عقب‌گرد با روش بررسی رو به جلو

فیلتر کردن: انتشار محدودیت (Constraint propagation)

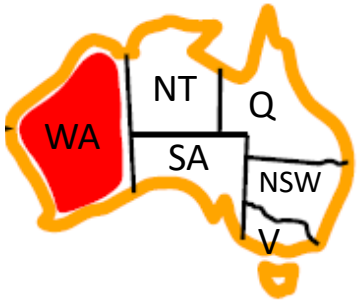
- بررسی رو به جلو، اطلاعات را از متغیرهای مقداردهی‌شده به متغیرهای مقداردهی‌نشده منتقل می‌کند، اما تمام شکست‌ها را نمی‌تواند در زودترین زمان ممکن تشخیص دهد.

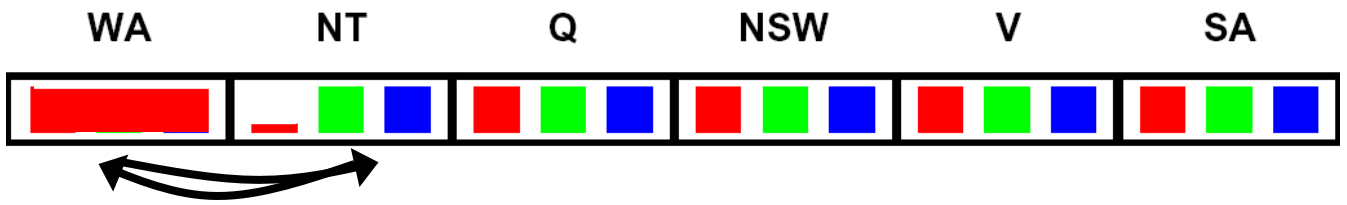


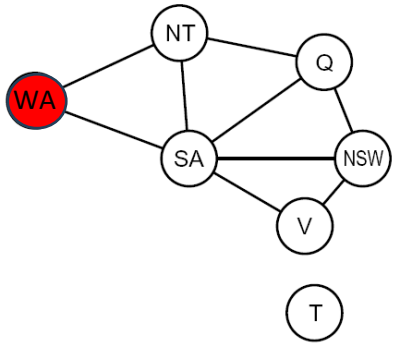
- NT و SA هر دو نمی‌توانند آبی باشند!
- انتشار محدودیت به طور مکرر بر سازگاری محدودیت‌ها به طور محلی تاکید دارد.

سازگاری کمان (یک کمان منفرد)

❑**سازگاری کمان:** کمان $X \rightarrow Y$ **سازگار** است اگر به ازای هر مقدار X حداقل یک مقدار Y وجود داشته باشد که با X سازگار باشد.



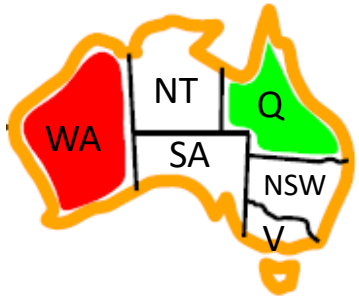


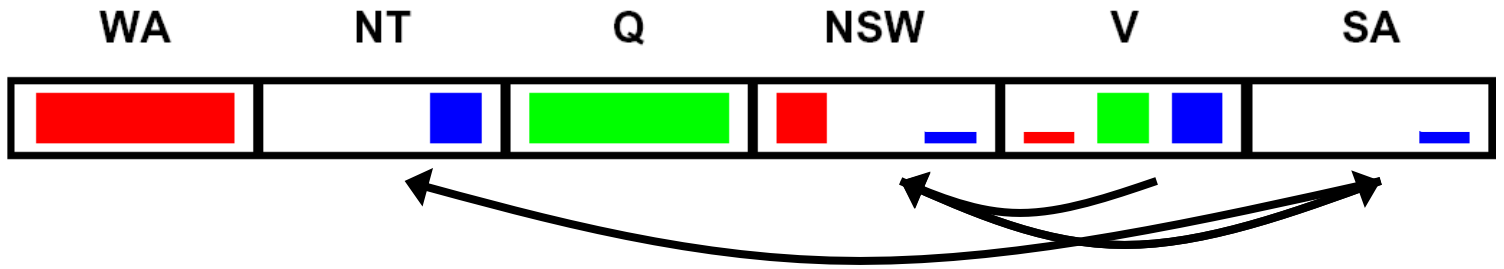


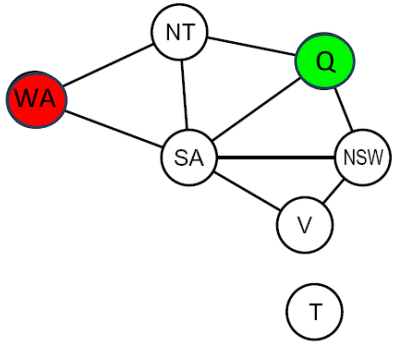
✓**همیشه از مبدأ (ابتدای یال) حذف انجام می‌شود!**

سازگاری کمان (یک CSP کامل)

❑**سازگاری کمان:** کمان $X \Rightarrow Y$ **سازگار** است اگر به ازای هر مقدار X حداقل یک مقدار Y وجود داشته باشد که با X سازگار باشد.







❑اگر مقداری از دامنه X حذف شود، همسایه‌های X نیاز به بررسی مجدد دارند، نه همه متغیرها

➢توجه. سازگاری کمانی شکست را زودتر از بررسی رو به جلو تشخیص می‌دهد.

➢می‌تواند به عنوان یک پیش‌پردازش اجرا شود یا پس از هر انتساب انجام شود.

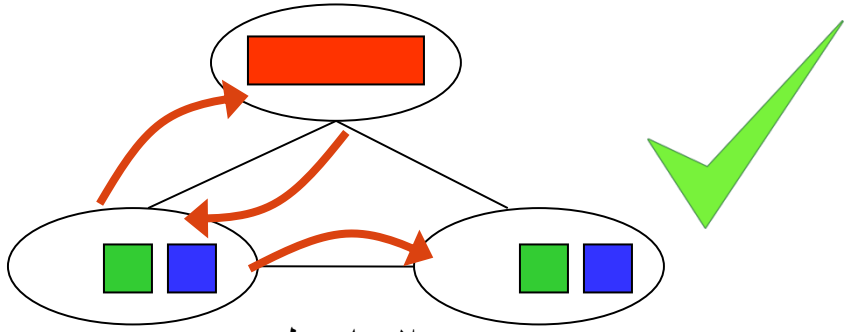
اعمال سازگاری کمانی در یک مسئله‌ی CSP

```
function AC-3( csp) returns the CSP, possibly with reduced domains
  inputs: csp, a binary CSP with variables {X1, X2, ..., Xn}
  local variables: queue, a queue of arcs, initially all the arcs in csp

  while queue is not empty do
    (Xi, Xj) ← REMOVE-FIRST(queue)
    if REMOVE-INCONSISTENT-VALUES(Xi, Xj) then
      for each Xk in NEIGHBORS[Xi] do
        add (Xk, Xi) to queue

---

function REMOVE-INCONSISTENT-VALUES( Xi, Xj) returns true iff succeeds
  removed ← false
  for each x in DOMAIN[Xi] do
    if no value y in DOMAIN[Xj] allows (x,y) to satisfy the constraint Xi ↔ Xj
      then delete x from DOMAIN[Xi]; removed ← true
  return removed
```

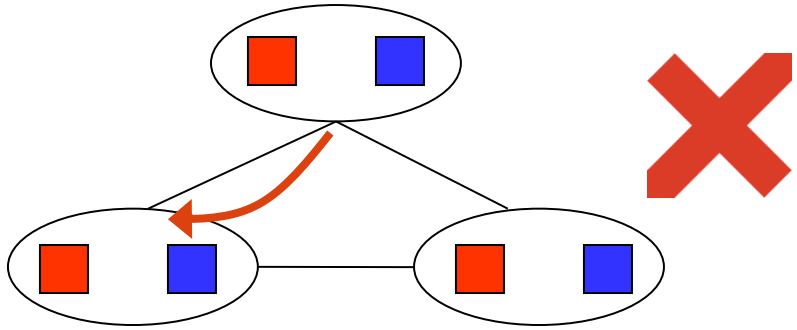
- زمان اجرا: $O(n^2d^3)$
  1. $O(n^2)$ تا کمان داریم
  2. (هر متغیر دارای d مقدار هست)، پس نهایتا d بار برای هر متغیر لازم می‌شود که کمان مجدد بررسی شود.
  3. هر بار بررسی کمان: هزینه $O(d^2)$

n: تعداد متغیر ها
d: دامنه متغیر ها

محدودیت‌های الگوریتم سازگاری کمانی



۲ راه حل



بدون راه حل

- **پس از اعمال سازگاری کمانی:**
  - ممکن است تنها یک راه‌حل باقی بماند.
  - ممکن است چندین راه‌حل باقی بماند.
  - ممکن است هیچ راه‌حلی باقی نماند (و متوجه آن نشویم).

- **سازگار کردن کمان‌ها همه شکست‌ها را تشخیص نمی‌دهد.**

ویدئوی نمایش رنگ‌آمیزی - جستجوی عقب‌گرد با روش بررسی رو به جلو

ویدئوی نمایش رنگ‌آمیزی - جستجوی عقب‌گرد با روش سازگاری کمان

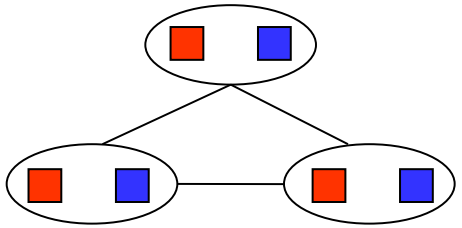
K- سازگاری

❑افزایش درجات سازگاری

- **۱- سازگاری (سازگاری گره):**
  - هر گره به‌تنهایی حداقل یک مقدار در دامنه‌اش دارد که محدودیت‌های تک‌متغیره (unary constraints) آن گره را ارضا می‌کند.

- **۲- سازگاری (سازگاری کمانی):**
  - برای هر زوج گره، اگر به یکی از آن‌ها مقداری اختصاص داده شود که با محدودیت‌ها سازگار باشد، می‌توان برای گره دیگر هم مقداری یافت که محدودیت‌های دوتایی بین آن‌ها را ارضا کند.

- **k- سازگاری:**
  - برای هر kگره، اگر به k−1گره مقداری داده شود که با هم سازگار باشند، می‌توان برای گره kام هم مقداری یافت که مجموعه‌ی کامل kتایی را سازگار کند.

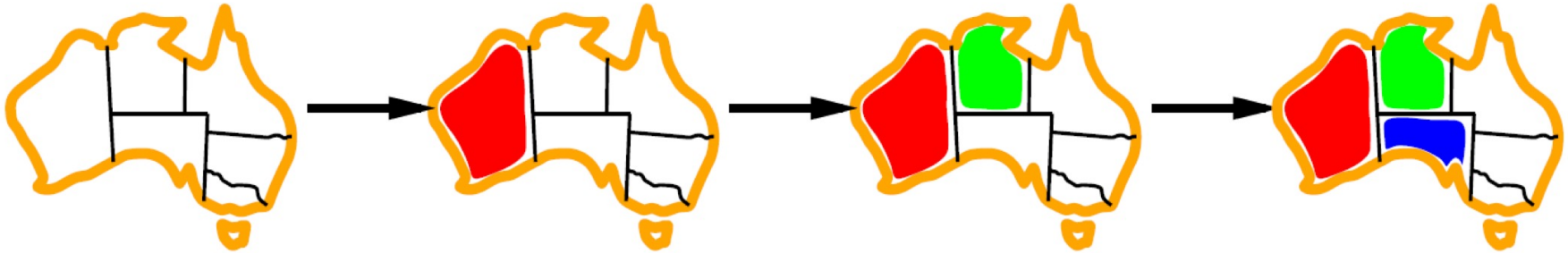


❑**نکته:** هر چه مقدار k بزرگ‌تر شود، بررسی سازگاری kتایی پرهزینه‌تر می‌شود.

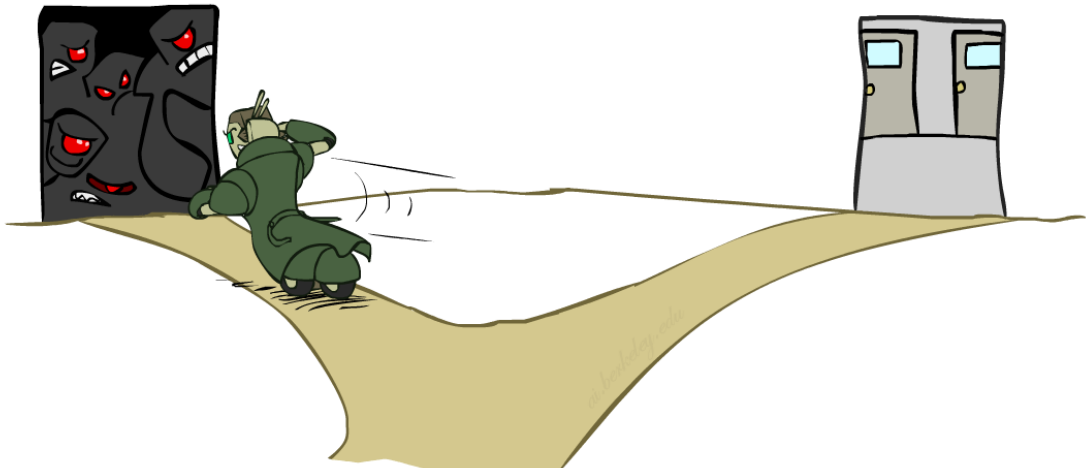
ترتیب‌دهی متغیرها (Ordering)

ترتیب‌دهی: کمترین مقادیر باقیمانده

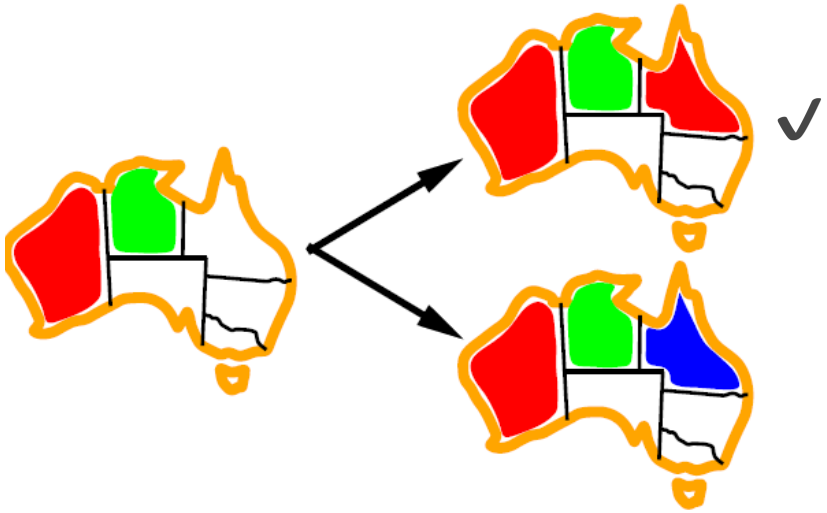
- ترتیب‌دهی متغیرها. کمترین مقادیر باقیمانده (MRV)
  - متغیری را انتخاب کن که کمترین مقادیر باقیمانده را دامنه خود دارد.



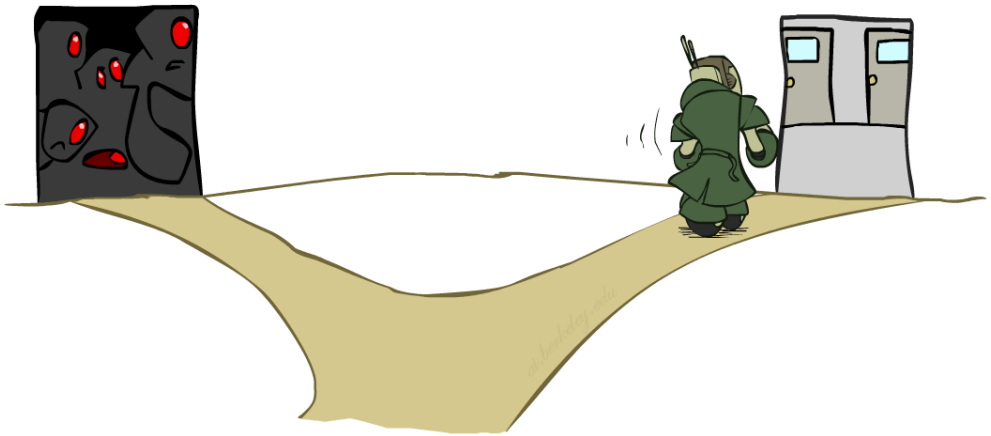
- چرا متغیر با کمترین مقادیر باقی مانده به متغیر با بیشترین مقادیر باقی مانده ارجح است؟
- نام دیگر: «متغیر با بیشترین محدودیت»
- ترتیب دهی «شکست- سریع»



ترتیب‌دهی: مقدار با کمترین محدودیت



- ترتیب‌دهی مقادیر. مقدار با کمترین محدودیت (LCV)
  - پس از انتخاب یک متغیر، مقداری را انتخاب کن که کمترین محدودیت را ایجاد می‌کند.
  - یعنی، مقداری که کمترین مقادیر را از دامنه متغیرهای باقیمانده حذف می‌کند.
  - تعیین چنین مقداری به اندکی محاسبات نیاز دارد (فیلتر کردن مجدد)



- با استفاده از ترتیب‌دهی مسئله ۱۰۰۰-وزیر قابل حل می‌شود.