به نام پروردگار دانایی

هوش مصنوعی

درس هشتم: مسائل ارضای محدودیت

سید کاوه احمدی

مسائل ارضای محدودیت (Constraint Satisfaction Problems)

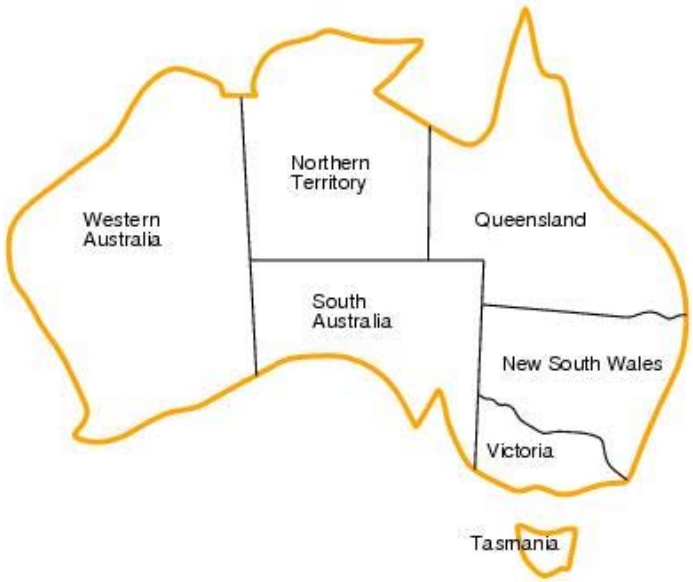
- در فصل‌های گذشته درباره مسائلی بحث شد که با ایده جستجو در فضای حالت حل می‌شوند.
  - جستجو تا رسیدن به یک حالت هدف ادامه پیدا می‌کند.
  - حالت‌ها فاقد یک ساختار داخلی مشخص بودند و صرفا از طریق روال‌هایی مانند تابع مابعد، اکتشافی و... بازنمایی می‌شدند.

مسائل ارضای محدودیت (CSP)

- به زبان رسمی مجموعه‌ای متناهی از
  - متغیرها $X_1, X_2, \ldots, X_n$ با دامنه نامتناهی برای هر متغیر $D_{X1}, D_{X2}, \ldots, D_{Xn}$
  - محدودیت‌ها $C_1, C_2, \ldots, C_m$
    - هر محدودیت C_i مشتمل بر برخی زیرمجموعه‌های متغیرها است که ترکیب‌های مجاز آن زیرمجموعه را معین می‌کند.
  - هر حالت با انتساب مقادیری به چند یا تمام متغیرها مشخص می‌شود.

مثال نقشه استرالیا

- می‌خواهیم مناطق درون نقشه را با رنگ‌های قرمز، سبز و آبی به گونه‌ای رنگ آمیزی کنیم که مناطق همجوار هم رنگ نباشند.



مثال نقشه استرالیا

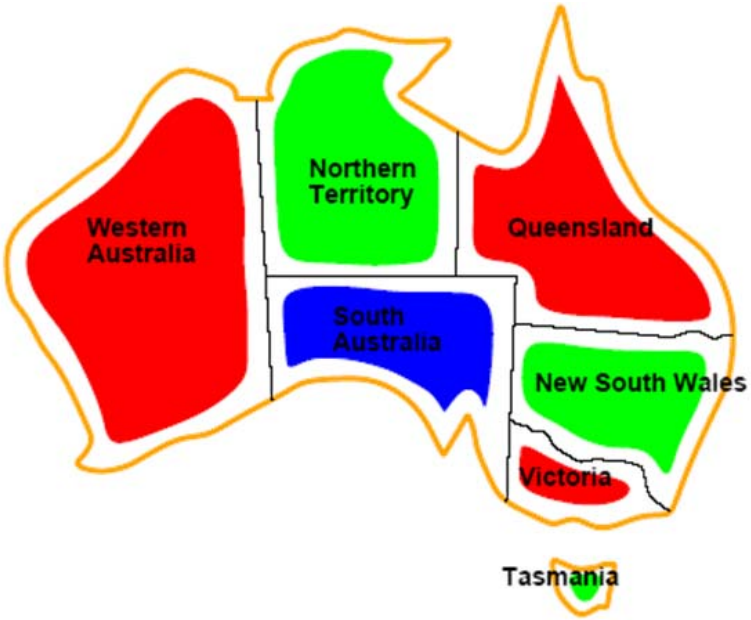
- فرموله کردن مساله بصورت CSP:
  - **متغیرها:** WA, NT, Q, NSW, V, SA, T
  - **دامنه:** D_i = {b, g, r} (برای تمام متغیرها)
  - **محدودیت‌ها:** دو منطقه مجاور، همرنگ نیستند
    - WA≠NT, WA≠SA, NT≠SA, NT≠Q, SA≠Q, SA≠NSW, SA≠V, Q≠NSW, NSW≠V
  - مثال: WA ≠ NT یعنی
    - **(WA,NT) ∈ {**(b, g), (b, r), (g, b), (g, r), (r, b), (r g)**}**

چند اصطلاح

- هر حالت با انتساب مقادیری به چند یا تمام متغیرها تعریف می‌شود.
  - **انتساب سازگار:** انتسابی که هیچ محدودیتی را نقض نکند.
  - **انتساب کامل:** انتسابی که همه متغیرها در آن حضور داشته باشد.
  - **راه حل مسئله CSP:** یک انتساب کامل و سازگار.
    - بعضی از CSPها به راه حل‌هایی نیاز دارند که تابع هدف را بیشینه کنند.

مثال نقشه استرالیا

- یک راه‌حل: یک انتساب کامل و سازگار



انواع محدودیت‌ها در مسائل CSP

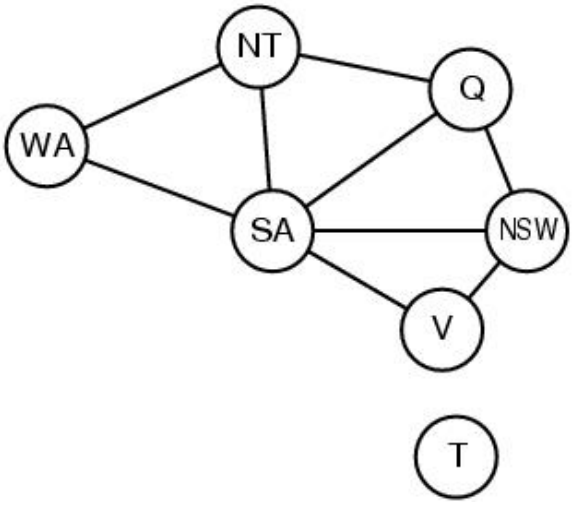
- محدودیت‌های کامل (Absolute)
  - مسائلی که در آن تمام محدودیت‌ها باید ارضا شود.
    - مثال: مسئله رنگ آمیزی نقشه
- محدودیت‌های اولویت (Preferences)
  - نشان می‌دهد که کدام راه‌حل‌ها ترجیح داده می‌شوند.
    - به هر یک از محدودیت‌ها هزینه‌ای داده می‌شود و باید هزینه را کم کنیم. (مانند مسائل جستجو می‌توان با آن برخورد کرد.)
    - مثال: زمانبندی وقت اساتید و اولویت های آنها.

انواع محدودیت‌ها در مسائل CSP

- محدودیت یکانی
  - محدودیت روی مقدار یک متغیر
  - به عنوان مثال در رنگ آمیزی نقشه WA رنگ سبز نباشد.
- محدودیت دودویی
  - محدودیتی که به دو متغییر مربوط باشد
  - به عنوان مثال SA≠NSW
  - CSP دودویی:آن است که فقط یک محدودیت دودویی دارد و می‌توانند مانند گراف مسئله رنگ آمیزی نقشه نمایش داده شود.
- محدودیت های مرتبه ی بالاتر
  - شامل ۳ یا چند متغیر.
  - اگر متغیرهای کمکی کافی معرفی شوند، هر محدودیت مرتبه‌ی بالاتر و با دامنه‌ی متناهی می‌تواند به مجموعه‌ای از محدودیت‌های دودویی کاهش یابد.

گراف محدودیت

- ترسیم گراف محدودیت‌ها به منظور نمایش محدودیت‌های مسئله و ساده‌تر کردن جستجو
  - گره‌ها: متغیرها
  - یال‌ها: محدودیت‌ها



خصوصیات CSP

- نمایش حالت‌ها در CSP از الگوی استانداردی پیروی می‌کند.
- برای CSP می‌توان یک فرموله‌بندی افزایشی (Formulation Incremental) ارائه کرد:

1. حالت اولیه: انتساب تهی $\{\}$ که در آن، هیچ متغیری مقدار ندارد.
2. تابع مابعد: انتساب یک مقدار به هر کدام از متغیرهای فاقد مقدار، به شرطی که با متغیرهایی که قبلا مقدار گرفتند، متضاد نباشند.
3. آزمون هدف: انتساب فعلی کامل است.
4. هزینه مسیر: هزینه ثابت برای هر مرحله.

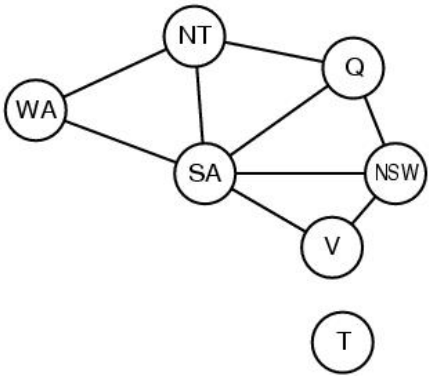
خصوصیات CSP

- چون نمایش حالات در CSP از الگوی مشخصی پیروی می‌کند (مجموعه‌ای از متغیرها و مقادیر منتسب به آنها):
  - تابع مابعد و آزمون هدف می‌تواند به صورت عمومی نوشته شود و برای تمامی CSPها مورد استفاده قرار گیرد.
  - می‌توانیم توابع کشف کننده‌های عمومی موثری را توسعه دهیم که نیاز به تبحر خاصی در مسئله هم نداشته باشند.
- یک CSP با ساختار درختی در زمان خطی برحسب تعداد متغیرها قابل حل است.

جستجوی عقبگرد (Backward) برای حل مسائل CSP

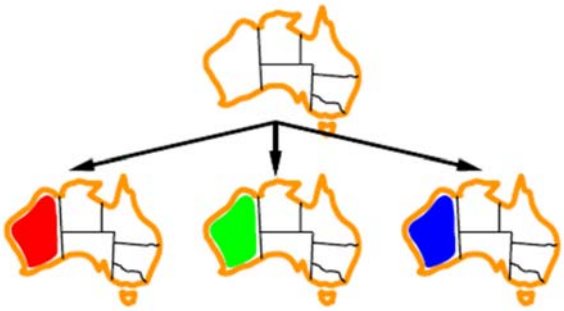
- جست و جوی عمقی
  - انتخاب مقادیر یک متغیر در هر سطح و عقبگرد در صورت عدم وجود مقداری مجاز برای انتساب به یک متغیر.
    - در این حالت فاکتور انشعاب از nd به d کاهش پیدا می‌کند
- یک الگوریتم ناآگاهانه است
  - برای مسئله های بزرگ کارآمد نیست
- خاصیت تعویض پذیری: مسئله وقتی تعویض پذیر است که ترتیب به کارگیری هر مجموعه‌ای از فعالیت‌ها تاثیری در نتیجه ندارد.

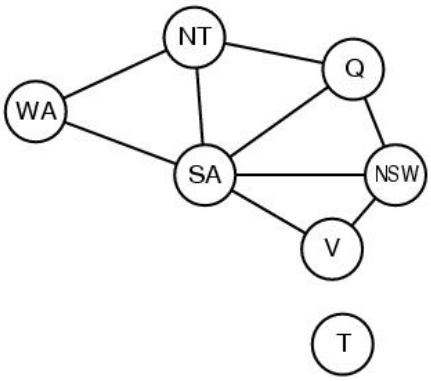
جستجوی عقبگرد برای رنگ‌آمیزی نقشه



شروع رنگ آمیزی از یک نقطه

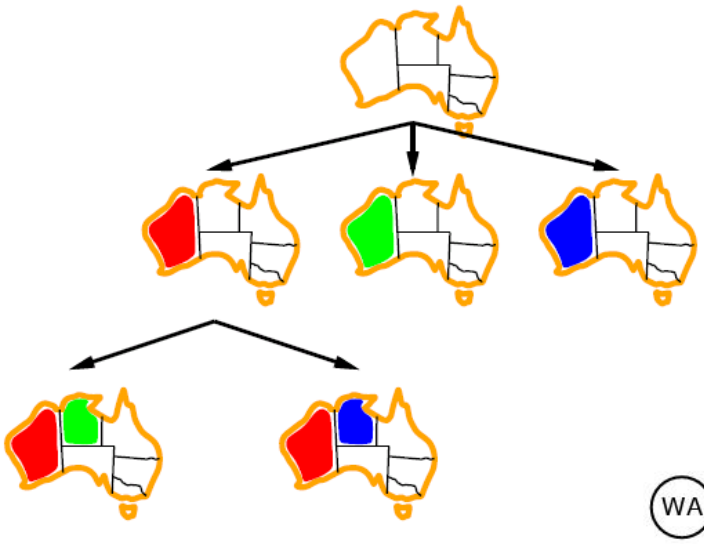
جستجوی عقبگرد برای رنگ‌آمیزی نقشه

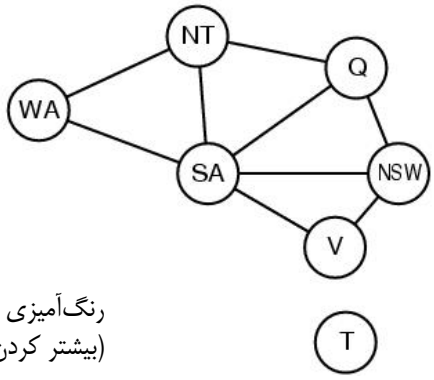




رنگ‌آمیزی WAدر سطح اول

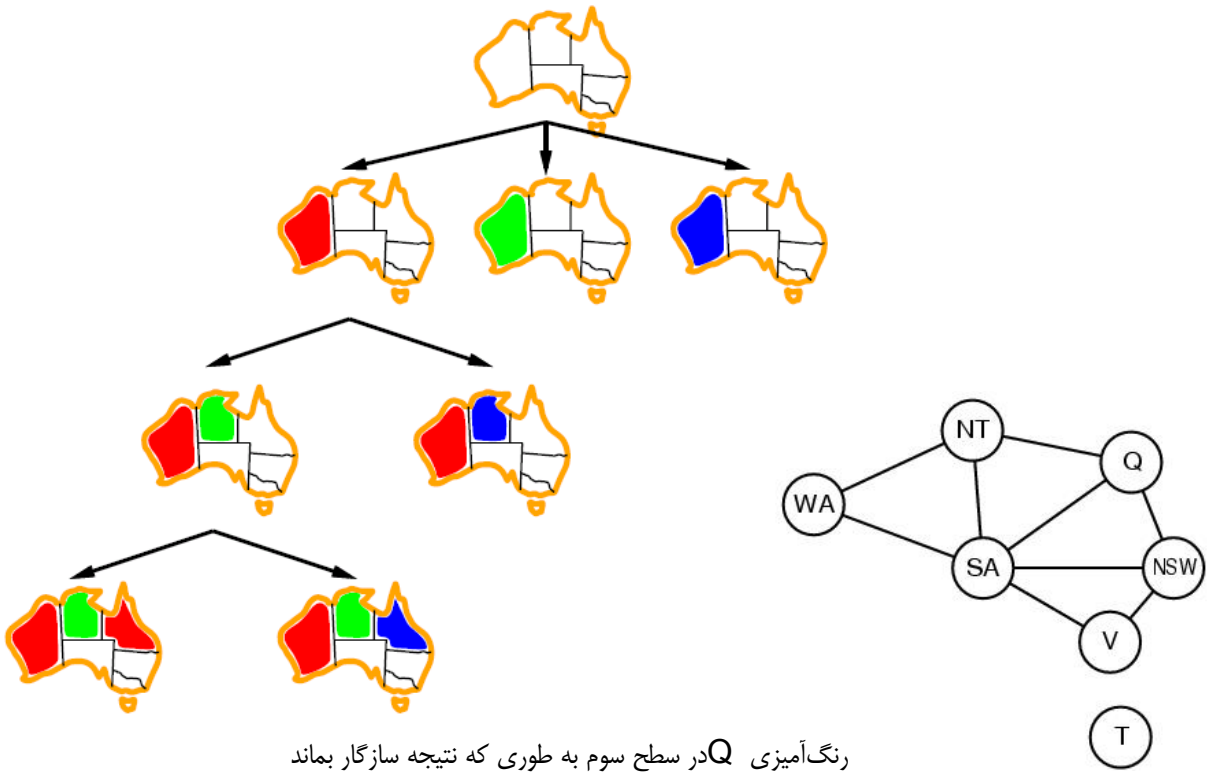
جستجوی عقبگرد برای رنگ‌آمیزی نقشه





رنگ‌آمیزی NT در سطح دوم به طوری که نتیجه سازگار بماند
(بیشتر کردن محدودیت‌هایی که باید اعمال شود به صورت افزایشی)

جستجوی عقبگرد برای رنگ‌آمیزی نقشه



رنگ‌آمیزی Qدر سطح سوم به طوری که نتیجه سازگار بماند

جستجوی عقبگرد برای رنگ‌آمیزی نقشه – ادامه

WA -> NT -> Q -> NS -> V -> SA -> T

آی تی ۸۷ و مشابه ۸۵

- کدام یک از مسائل زیر برای استفاده از الگوریتم ارضای محدودیت ها مناسب نیست؟

1. جورچین ۸
2. زمان بندی امتحانها
3. حل جدول کلمات متقاطع
4. چینش ساختارهای منطقی بر روی یک تراشه

آی تی ۸۷ و مشابه ۸۵

- گزینه ۱ پاسخ است:
  - در مسائل CSP هزینه مسیر تعیین نمی‌شود.
  - کاربردهای متفاوت CSP در گزینه‌ها آمده است!

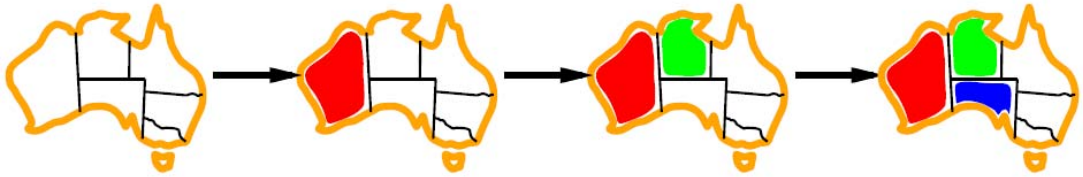
ترتیب انتخاب متغیرها در CSP

- ترتیب مقداردهی به متغیرها در مسائل CSP می‌تواند در کارایی جستجو تاثیرگذار باشد.
  - بعد از جایگذاری WA = r و NT = g تنها یک مقدار برای SA باقی می‌ماند و اگر ابتدا آنرا انتخاب کنیم، پی جویی به عقب زودتر رخ می‌دهد.

ترتیب انتخاب متغیرها در CSP

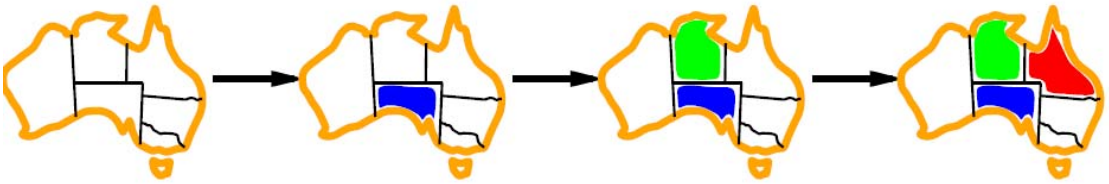
- از دو روش کلی می توان بهره جست:
  1. کمترین مقادیر باقیمانده (Minimum Remaining Value - MRV)
  2. اکتشاف درجه‌ای (Degree Heuristic)
- استفاده از هر دو روش به صورت توأم مناسب‌ترین گزینه است. (روش اول در اولویت است.)

ترتیب انتخاب متغیرها: کمترین مقادیر باقیمانده (MRV)



- متغیری انتخاب می‌شود که به احتمال زیاد به زودی با شکست مواجه شده و درخت جستجو را هرس می‌کند (Fail First)
  - انتخاب متغیری با کمترین مقادیر معتبر
  - اگر متغیر X فاقد مقادیر معتبر باشد، اکتشاف MRV متغیر X را انتخاب می کند و با شکست مواجه می شود.
  - با نام های اکتشاف "محدودترین متغیر" یا "اولین شکست"نیز شناخته می شود.
- در انتخاب اولین ناحیه‌ای که باید رنگ شود کمکی نمی‌کند، زیرا ابتدا هر ناحیه سه رنگ معتبر دارد. در این مورد باید از اکتشاف درجه‌ای استفاده کرد.

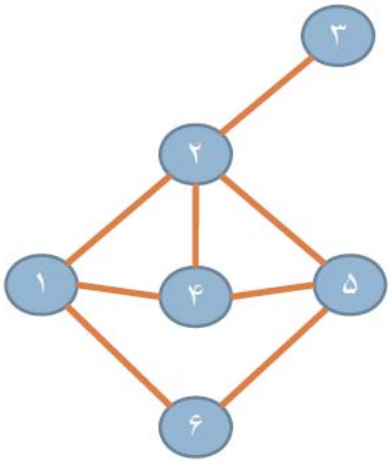
ترتیب انتخاب متغیرها: اکتشاف درجه‌ای (DH)



- سعی می‌کند فاکتور انشعاب را در انتخاب آینده کم کند.
- متغیری را انتخاب می کند که بیشترین محدودیت‌ها را روی متغیرهای بدون انتساب ایجاد می‌کند.
  - متغیری که بیشترین درجه را در گراف محدودیت داشته باشد.
- در ابتدای مسأله یا زمانی که MRV برای همه‌ی متغیرها، به تعداد یکسان حالت در نظر می‌گیرد، از DH استفاده می‌شود.

مهندسی ۸۷

- در صورتی که بخواهیم با استفاده از روش ارضای محدودیتها گراف مقابل را به سه رنگ، رنگ آمیزی نماییم، پس از رنگ آمیزی رئوس ۱ و ۲ بهتر است کدام رأس رنگ آمیزی شود؟



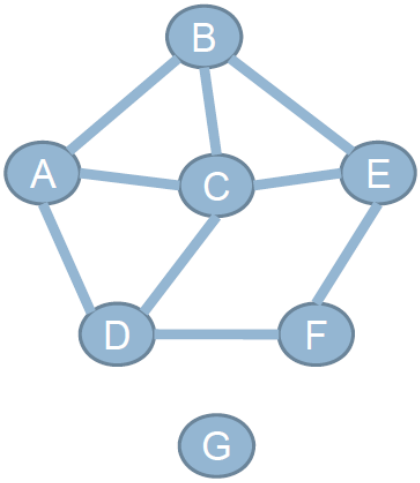
1. ۵
2. ۶
3. ۳
4. ۴

مهندسی ۸۷

- پاسخ گزینه ۴ است.
  - پس از رنگ آمیزی گره‌های ۱ و ۲، با توجه به وجود ۳ رنگ برای رنگ آمیزی و با در نظر گرفتن محدودیت‌ها، گره ۳ دو، گره ۴ یک، گره ۵ دو و گره ۶ نیز دو رنگ ممکن برای رنگ آمیزی خواهند داشت. براساس MRV گره با کمترین مقدار باقی‌مانده اول رنگ می‌شود.

مکاترونیک ۸۶

- در یک مسأله می خواهیم نقشه را با سه رنگ، رنگ آمیزی کنیم، طوری که هیچ دو کشوری که با هم مرز مشترک دارند همرنگ نشوند. اطلاعات نقشه را به صورت گراف محدودیت زیر نمایش داده ایم. کدام گزینه، ترتیب بهتری برای دو کشوری که اول انتخاب می شوند، از چپ به راست است؟



1. C,A
2. C,F
3. G,B
4. G,F

مکاترونیک ۸۶

- گزینه ۱ پاسخ است.

  - در ابتدای جستجو MRV کمکی به ما نمی‌کند چراکه همه متغیرها می‌توانند ۳ مقدار بگیرند. براساس DH گره‌ی که بیشترین محدودیت را اعمال می‌کند اول انتخاب می‌شود.
  - در اینجا C روی ۴ گره دیگر محدودیت ایجاد کرده و اول انتخاب می‌شود.
  - در بین گزینه ۱ و ۲ که می‌تواند پاسخ باشد، پس از رنگ آمیزی C هر دو دو رنگ باقی مانده خواهند داشت که باز MRV به ما کمک نمی‌کند. براساس DH نیز A روی ۳ گره در حال اعمال محدودیت است (به عبارت بهتری ۲ گره رنگ نشده)

مهندسی ۸۴

- درمسائل ارضای محدودیت ها یک تابع مکاشفه ای مناسب، برای انتخاب متغیر بعدی برای مقدار دهی کدام است؟

1. انتخاب متغیری با بزرگترین دامنهٔ مقادیر مجاز
2. انتخاب متغیری با کوچکترین دامنهٔ مقادیر مجاز
3. انتخاب متغیری که کمترین میزان مقادیر مجاز را از دامنهٔ سایر متغیرها حذف کند.
4. انتخاب متغیری که بیشترین میزان مقادیر مجاز را از دامنهٔ سایر متغیرها حذف کند.

مهندسی ۸۴

- گزینه ۲: مطابق MRV و DH

ترتیب انتساب مقادیر به متغیرها

- برای انتساب مقدار به متغیر انتخاب شده از مراحل قبل از روش اکتشاف مقداری با کمترین محدودیت (Least Constraining Value) کمک گرفته می شود
  - این روش مقداری را ترجیح می‌دهد که در گراف محدودیت، متغیرهای همسایه به ندرت آن را انتخاب می‌کنند.
  - تلاش بر ایجاد بیشترین قابلیت انعطاف برای انتساب بعدی متغیرهاست.

یک مقدار برای SA باقی می‌ماند.

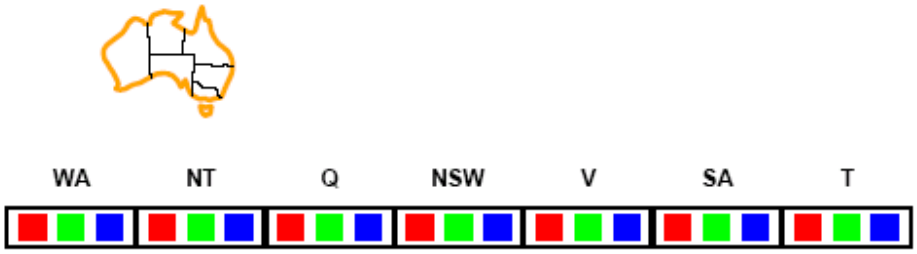
مقداری برای SA باقی نمی‌ماند.

انتشار اطلاعات مابین محدودیت‌ها

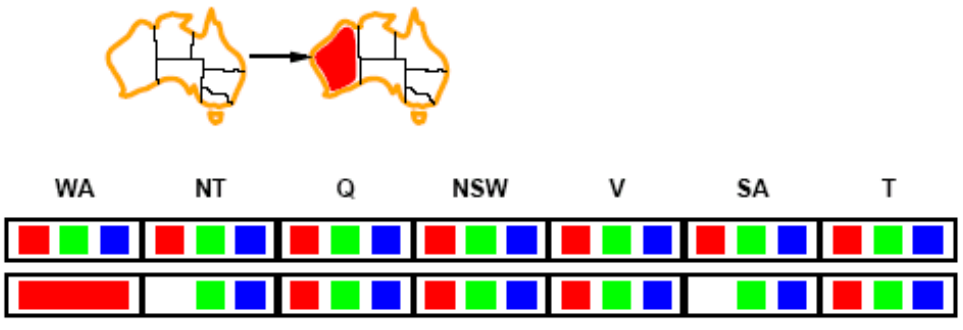
- تاکنون در الگوریتم‌ها، محدودیت‌ها فقط در زمان انتخاب متغیر بعدی در نظر گرفته شد، اما اگر بعضی محدودیت‌ها قبل از جستجو در نظر گرفته شود، فضای جستجو کوچکتر می‌شود.
  - بدین منظور، از چند الگوریتم استفاده می‌شود:
    1. بررسی پیشرو – کنترل رو به جلو (Forward Checking)
    2. انتشار محدودیت (Constraint Propagation)

بررسی پیشرو (Forward Checking)

- وقتی انتساب به X صورت می‌گیرد، فرآیند بررسی پیشرو، متغیرهای بدون انتساب مثل Y را در نظر می‌گیرد که از طریق یک محدودیت به X متصل است و هر مقداری را که با مقدار انتخاب شده برای X برابر است، از دامنه Y حذف می‌کند

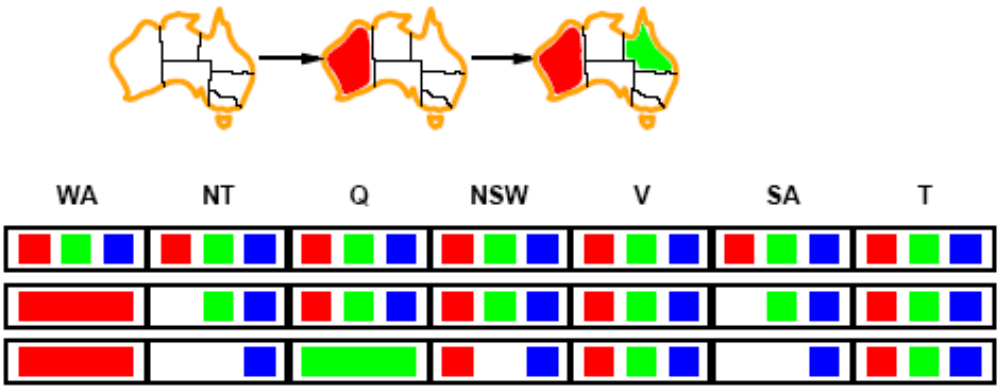


بررسی پیشرو (Forward Checking)



انتساب رنگ قرمز به WA و در نتیجه حذف رنگ قرمز از دامنه‌ی نواحی هم جوار با WA یعنی NT و SA

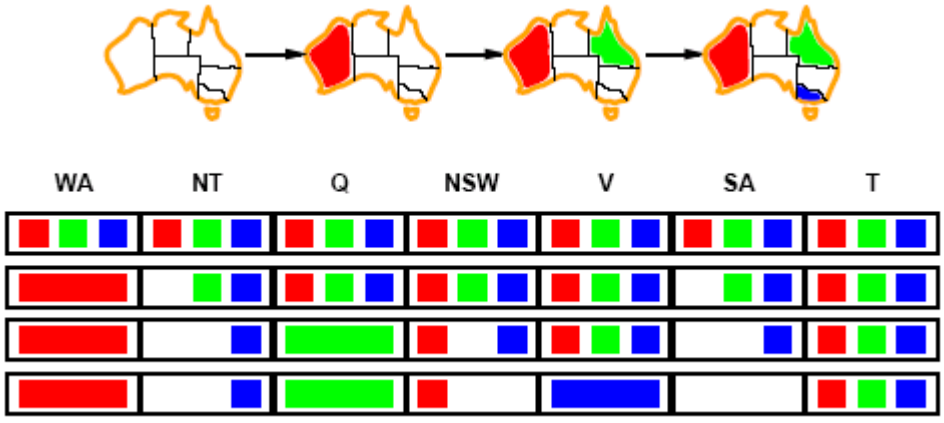
بررسی پیشرو (Forward Checking)



فرض کنید Q به عنوان گره بعدی انتخاب می‌شود.

دامنه‌ی NT و SA شامل یک مقدار است.

بررسی پیشرو (Forward Checking)



با انتساب V=blue، دامنه‌ی SA خالی می‌شود لذا بررسی پیشرو در می‌یابد که انتساب جزئی {WA=red, Q=green, V=blue} با محدودیت‌های مسئله سازگار نیست و الگوریتم فورا برگشت می کند.

بررسی پیشرو (Forward Checking)

- با انتخاب مقدار برای هر منطقه، سایر مناطق محدودتر می‌شوند.
- بررسی پیشرو یک روش کارآمد برای محاسبه‌ی اطلاعات مورد نیاز هیوریستیک MRV است.
- این جستجو وجود تناقض را سریعتر از جستجوی عقبگرد ساده تشخیص می‌دهد.

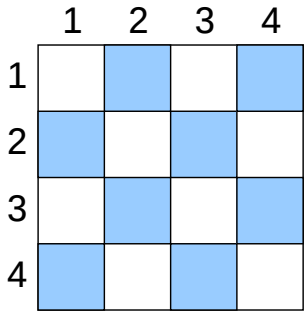
بررسی پیشرو(مثال ۴ وزیر)

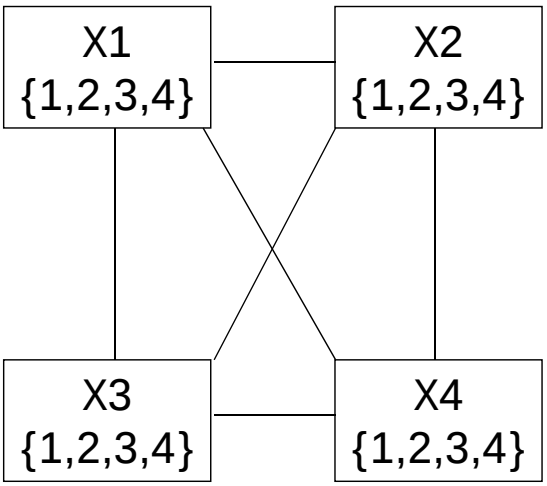
- تعریف پارامترها

- 4 variables X_i, i = 1 to 4
- Domain for each variable {1, 2, 3, 4}
- Constraints are of the forms:
  - $X_i = k \rightarrow X_j \neq k$ for all j = 1 to 4, j≠i
  - $X_i = k_i, X_j = k_j \rightarrow |i-j| \neq |k_i - k_j|$
    - for all j = 1 to 4, j≠i

بررسی پیشرو(مثال ۴ وزیر)

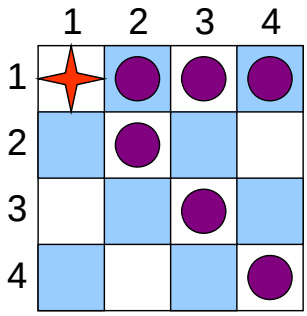
- هر وزیر در یک ستون

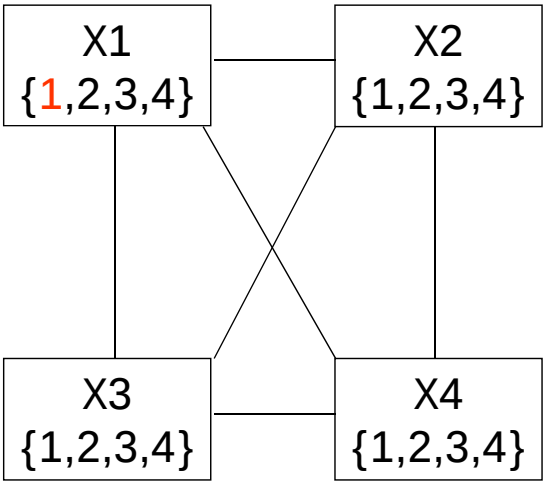




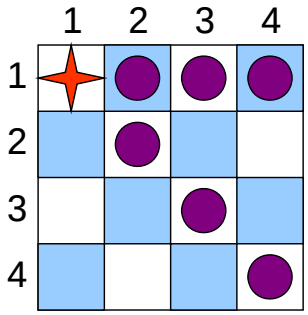
[4-Queens slides copied from B.J. Dorr]

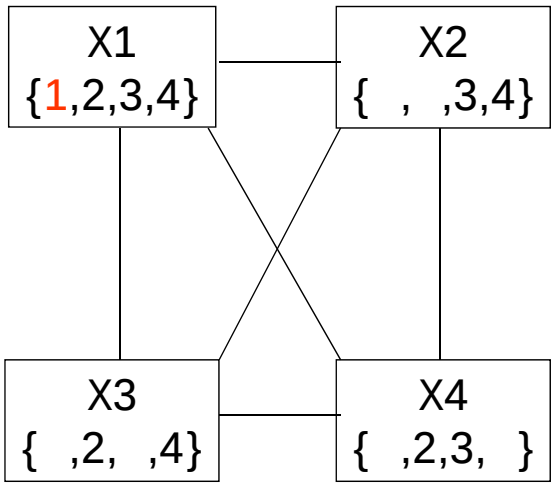
بررسی پیشرو(مثال ۴ وزیر)



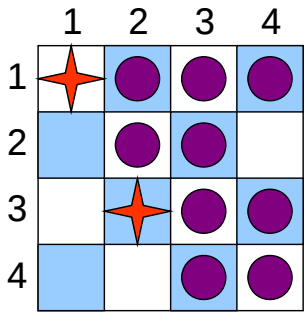


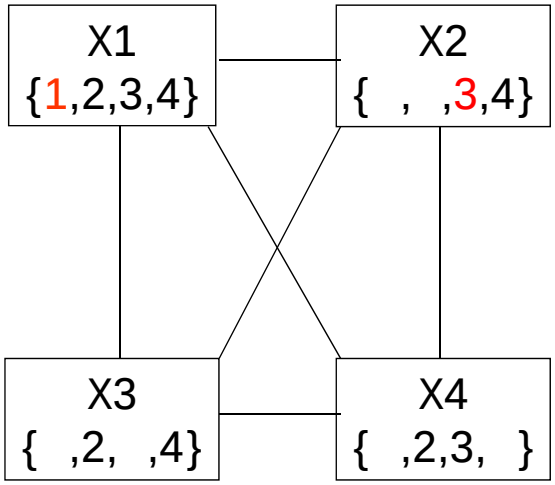
بررسی پیشرو(مثال ۴ وزیر)



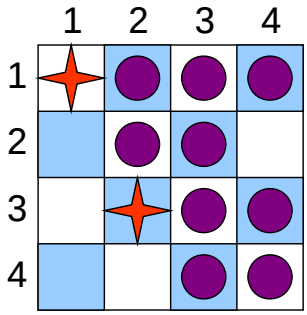


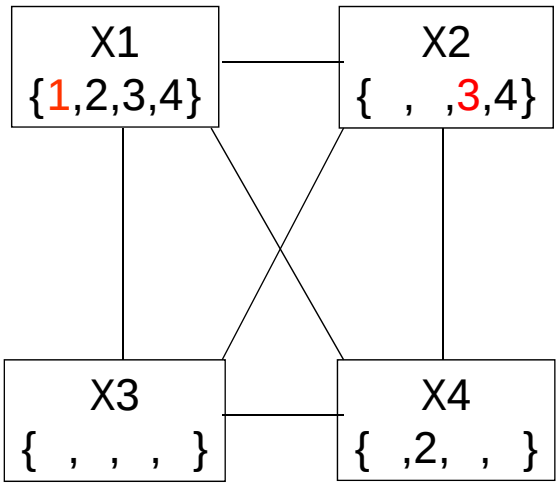
بررسی پیشرو(مثال ۴ وزیر)



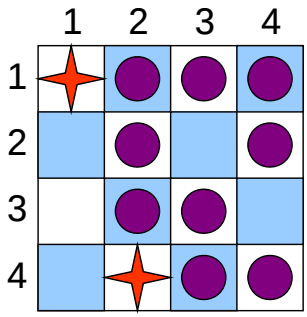


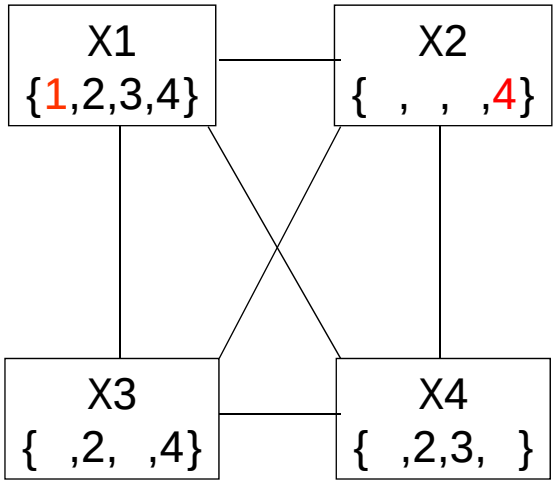
بررسی پیشرو(مثال ۴ وزیر)



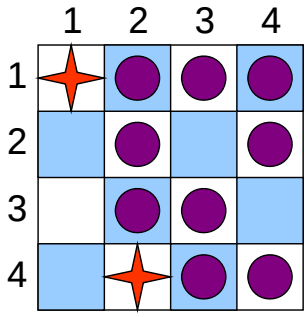


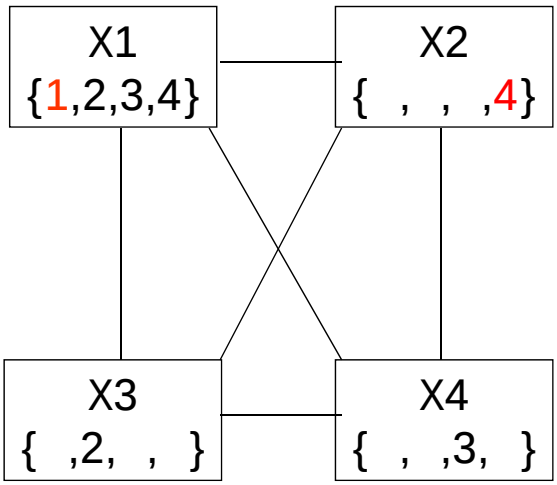
بررسی پیشرو(مثال ۴ وزیر)



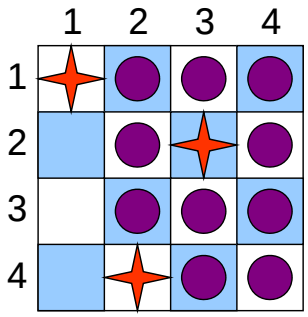


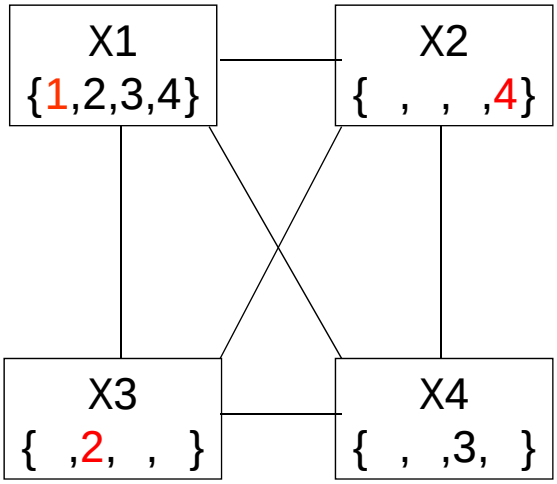
بررسی پیشرو(مثال ۴ وزیر)



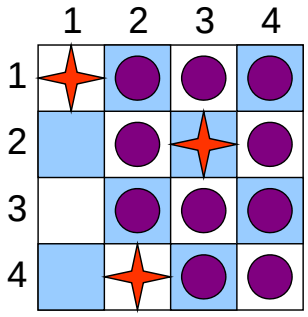


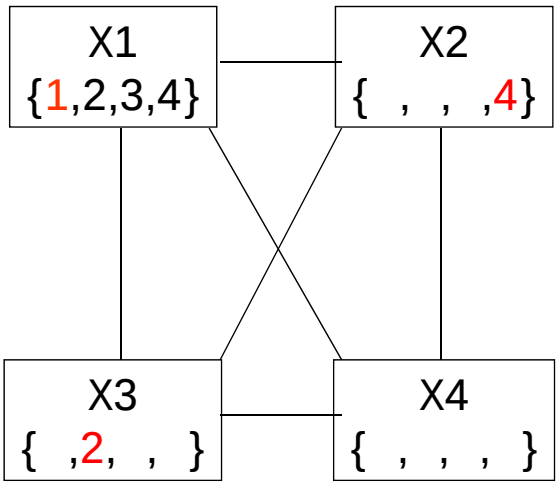
بررسی پیشرو(مثال ۴ وزیر)



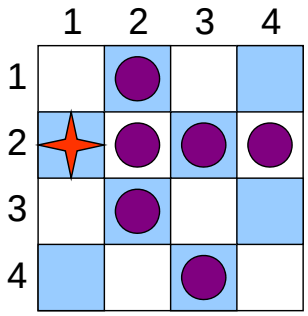


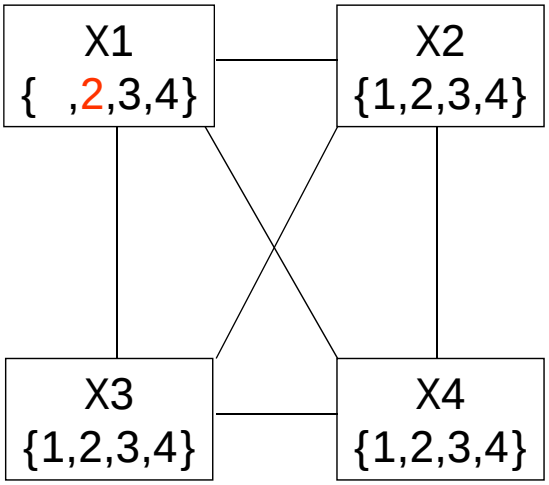
بررسی پیشرو(مثال ۴ وزیر)



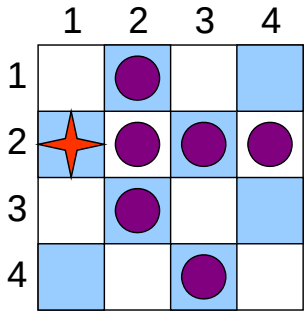


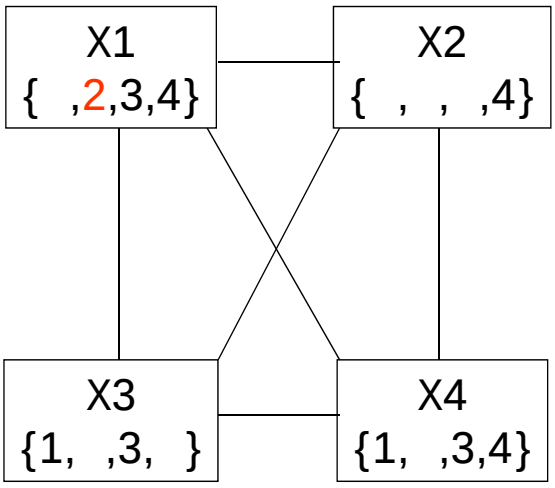
بررسی پیشرو(مثال ۴ وزیر)



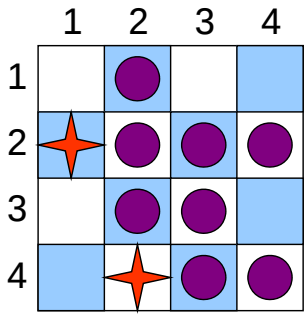


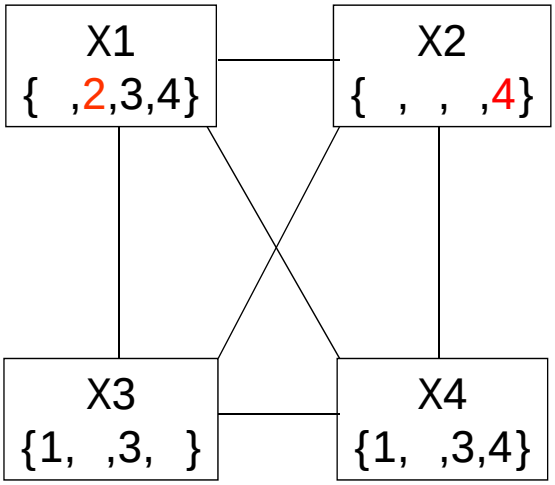
بررسی پیشرو(مثال ۴ وزیر)



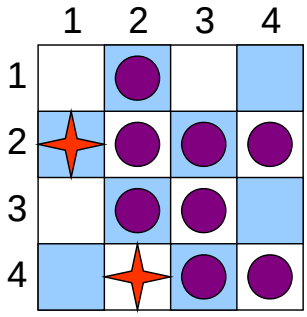


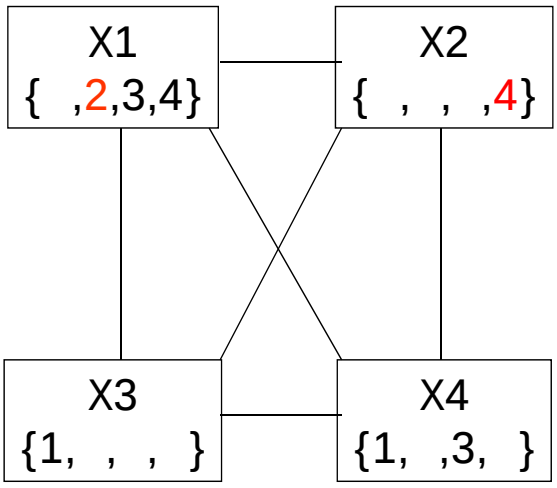
بررسی پیشرو(مثال ۴ وزیر)



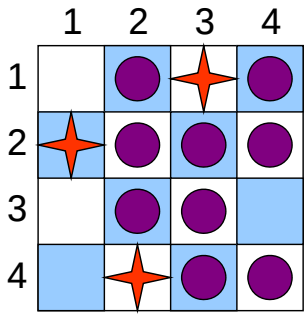


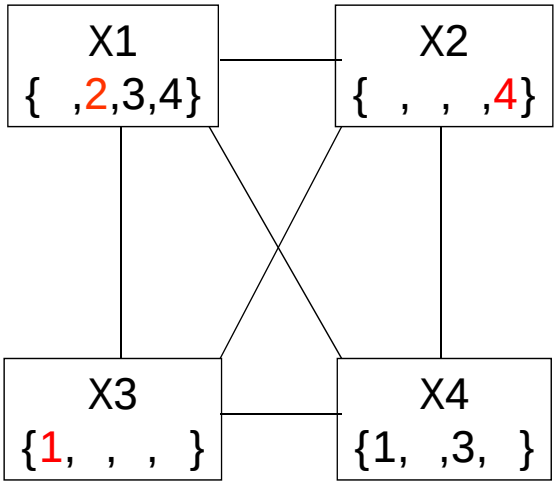
بررسی پیشرو(مثال ۴ وزیر)



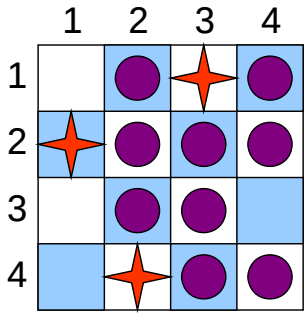


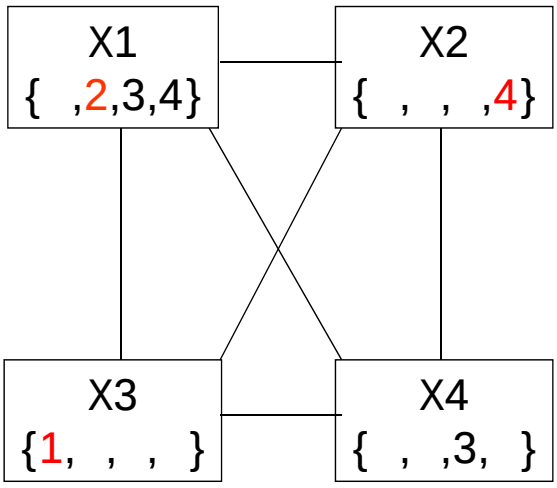
بررسی پیشرو(مثال ۴ وزیر)



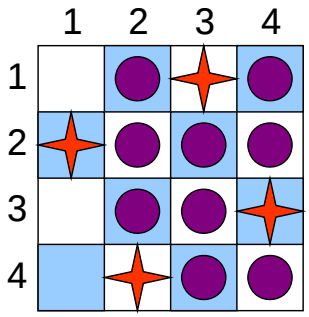


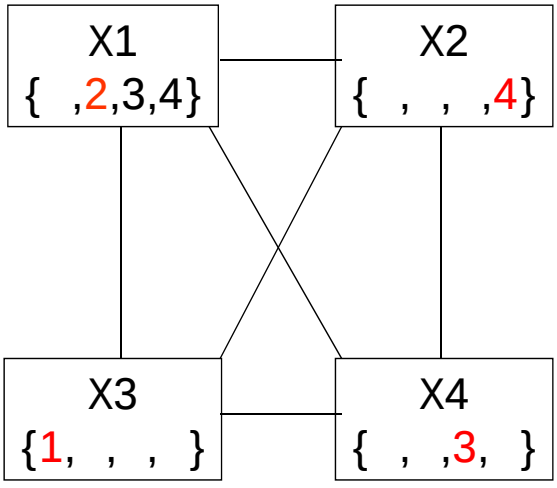
بررسی پیشرو(مثال ۴ وزیر)





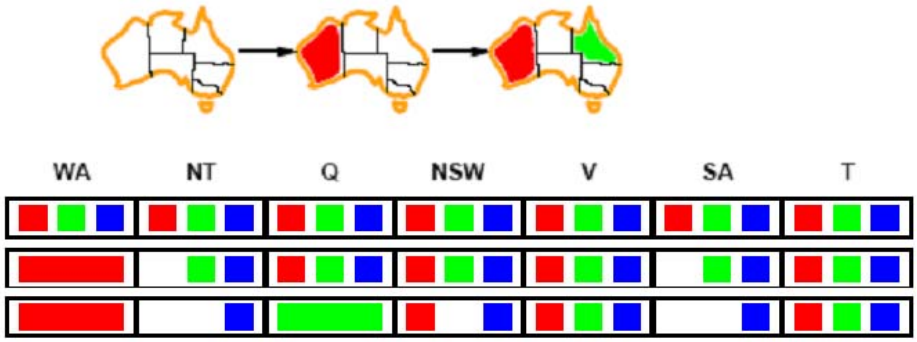
بررسی پیشرو(مثال ۴ وزیر)





انتشار محدودیت (Constraint Propagation)

- بررسی پیشرو بسیاری از ناسازگاری‌ها را تشخیص می‌دهد ولی قادر به کشف تمامی آنها نیست.
  - در مثال قبل با وجود آنکه NT و SA همسایه هستند و مقدار هردو فقط می‌تواند آبی باشد، تناقض تشخیص داده نشد.



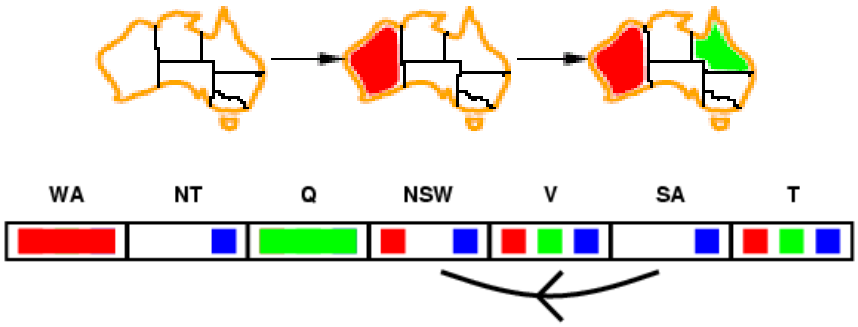
انتشار محدودیت (Constraint Propagation)

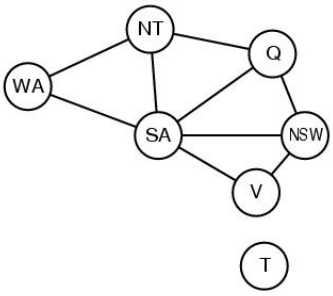
- ایده: پخش محدودیت‌های یک متغیر به متغیرهای دیگر
- مثال: پخش محدودیت‌های WA و Q به NT و SA
- بررسی پیشرو نتوانست این مورد را به عنوان یک ناسازگاری کشف کند چراکه نگاه رو به جلو ندارد.

سازگاری یال (Arc Consistency)

- روشی سریع برای پخش محدودیت و قویتر از بررسی پیشرو
  - در هر مرحله در نودهایی که تغییری صورت گرفته باشد، سازگاری با همسایه‌ها بررسی می‌شود.
- Arc: یال جهت‌دار در گراف محدودیت
- بررسی سازگاری یال
  - یک مرحله پیش پردازش، قبل از شروع جستجو
  - یک مرحله پخشی پس از هر انتساب هنگام جستجو (MAC)

سازگاری یال

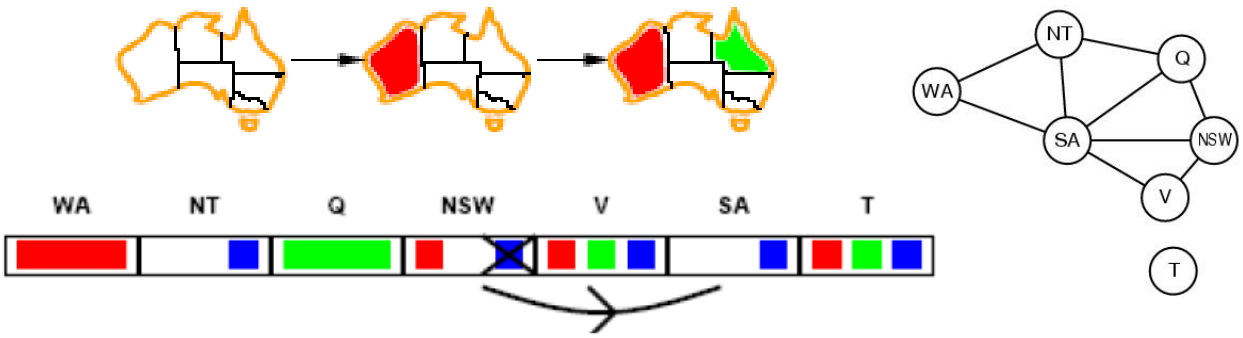




- SA → NSW سازگار است اگر
  - SA = blue → NSW = red

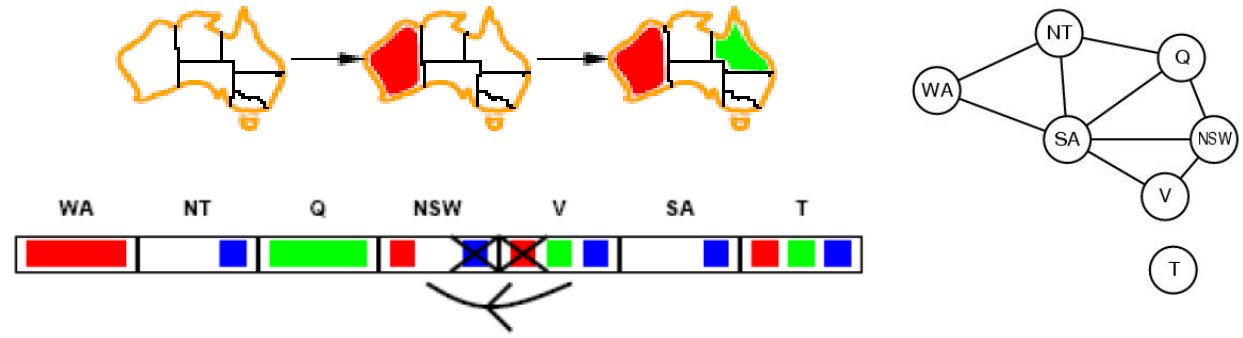
for every value x of X there is some allowed y

سازگاری یال



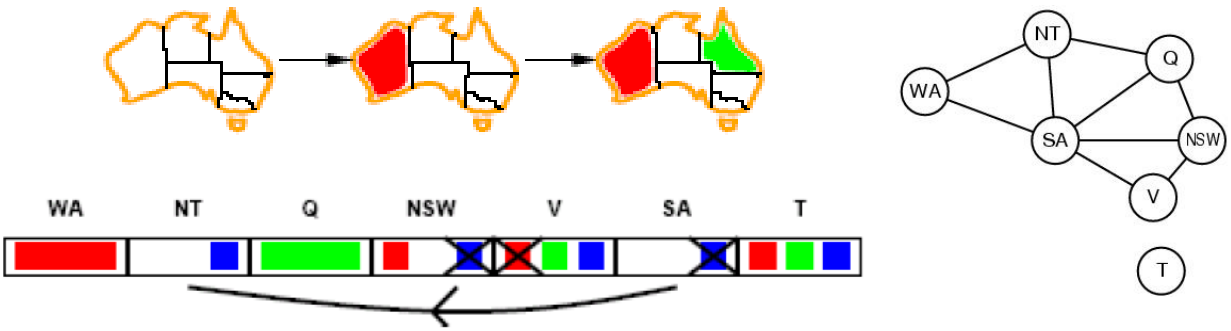
- NSW → SA سازگار نیست چون:
  - NSW = red → SA = blue
  - NSW = blue → SA = ???
- یال می‌تواند با حذف blue از NSW سازگار شود.

سازگاری یال



- V → NSW سازگار نیست چون:
  - V = red → NSW = ???
- یال می‌تواند با حذف red از V سازگار شود.

سازگاری یال



- SA → NT سازگار نیست
  - SA = blue → NT = ???
- یال می‌تواند با حذف blue از SA سازگار شود. ولی دیگر هیچ انتخابی برای SA باقی نمی‌ماند و الگوریتم مجبور به بازگشت می‌شود.

If X loses a value, neighbors of X need to be rechecked

سازگاری K (K-Consistency)

- سازگاری یال تمام ناسازگاری‌های ممکن را مشخص نمی‌کند.
- با روش سازگاری K، شکل‌های قویتری از پخش را می‌توان تعریف کرد.
  - در صورتی CSP سازگاری K دارد که برای هر K-1 متغیر و انتساب سازگار با آن متغیرها، همیشه بتواند یک مقدار سازگار به متغیر Kام نسبت داده شود.

سازگاری K

- بطور مثال:
  - سازگاری ۱: هر متغیر با خودش سازگار است (Node Consistency)
  - سازگاری ۲: همان سازگاری یال (Arc Consistency)
  - سازگاری ۳ : بسط هر جفت از متغیرهای همجوار به سومین متغیر همسایه (Path Consistency)
- گراف در صورتی قویاً سازگار K است که:
  - سازگار K باشد
  - همچنین سازگار K-1، K-2 و ... سازگار ۱ باشد.
- در این صورت، مسأله را بدون عقبگرد می‌توان حل کرد.
- پیچیدگی زمانی آن O(nd)است.

Arc Consistency Algorithm AC-3

function AC-3(*csp*) **returns** false if an inconsistency is found and true otherwise
  inputs: *csp*, a binary CSP with components (X, D, C)
  local variables: *queue*, a queue of arcs, initially all the arcs in *csp*

  while *queue* is not empty **do**
    $(X_i, X_j) \leftarrow$ REMOVE-FIRST(*queue*)
    if REVISE(*csp*, X_i, X_j) **then**
      if size of $D_i = 0$ **then return** *false*
      for each X_k **in** X_i.NEIGHBORS - $\{X_j\}$ **do**
        add (X_k, X_i) to *queue*
  return *true*

---

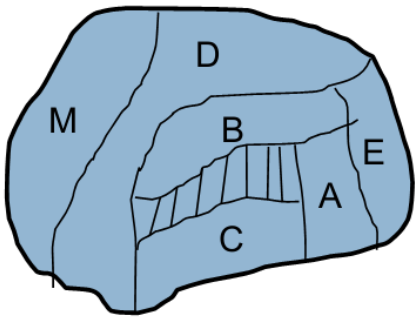
function REVISE(*csp*, X_i, X_j) **returns** true iff we revise the domain of X_i
  revised $\leftarrow$ *false*
  for each x **in** D_i **do**
    if no value y in D_j allows (x,y) to satisfy the constraint between X_i and X_j **then**
      delete x from D_i
      revised $\leftarrow$ *true*
  return *revised*

Time complexity: $O(n^2d^3)$

مهندسی ۸۹

- در یک مسألهٔ رنگ آمیزی نقشه به شکل زیر می خواهیم هیچ دو کشور همسایه‌ای همرنگ نباشند و هر کشور به سه رنگ قرمز و سبز و آبی می تواند رنگ شود. فرض کنید در یک جستجوی ارضای محدودیت برای این مسأله، به کشور A رنگ قرمز و به کشور C رنگ آبی نسبت داده باشیم. حال اگر بخواهیم به کشور M رنگ قرمز را نسبت دهیم، کدام تست زیر بروز تناقض و نیاز به عقبگرد را پیشبینی می کند؟

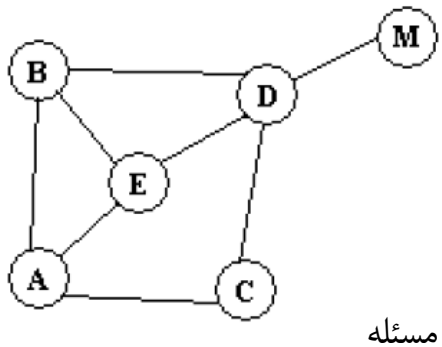
1. Forward Checking
2. Path Consistency
3. Node Consistency
4. تناقضی کشف نخواهد شد



۲

مهندسی ۸۹

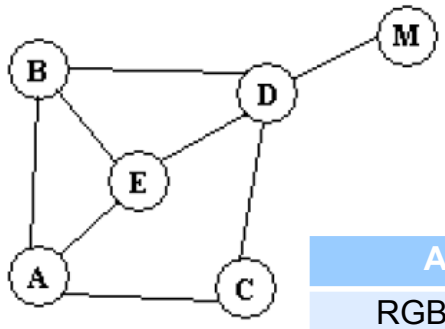
- گراف محدودیت مسئله به شکل زیر است:



  - A قرمز است و C آبی. اگر بخواهیم M قرمز شود:
    - D فقط می‌تواند سبز باشند.
    - B و E نیز فقط می‌توانند سبز یا آبی باشند.
  - با توجه به اعمال محدودیت D، B و E روی همدیگر، مسئله جوابی ندارد (دو رنگ داریم و سه گره)
  - سوال این است که کدام روش شکست را زودتر تشخیص می‌دهد.

مهندسی ۸۹

- بررسیِ بررسی پیشرو:

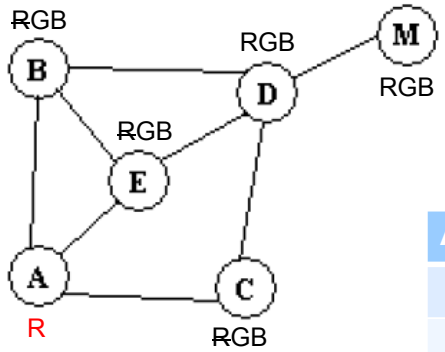


A	B	C	D	E	M
RGB	RGB	RGB	RGB	RGB	RGB
R	GB	GB	RGB	GB	RGB
R	GB	**B**	RG	GB	RGB
R	GB	B	G	GB	**R**
R	**B**	B	G	G	R

E نمی‌تواند سبز باشد و این مورد توسط بررسی پیشرو تشخیص داده نشده است.

مهندسی ۸۹

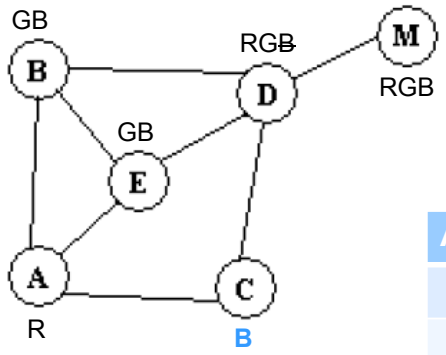
- بررسی سازگاری K
  - K=2



Arc examined	Value deleted
A-B	Red from B
A-C	Red from C
A-E	Red from E
B-D	None
B-E	None
C-D	None
E-B	None
E-D	None

مهندسی ۸۹

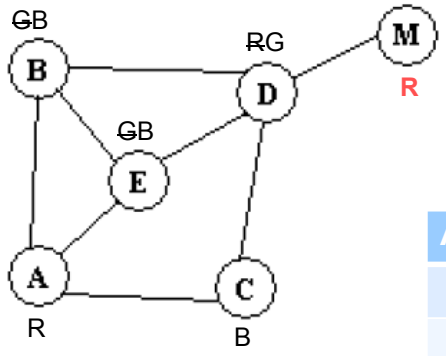
- بررسی سازگاری K
  - K=2



Arc examined	Value deleted
C-D	Blue from D
D-B, D-C, D-E, D-M	None

مهندسی ۸۹

- بررسی سازگاری K
  - K=2



Arc examined	Value deleted
M-D	Red from D
D-B	Green from B
D-E	Green from E
…	…
B-E	????

مهندسی ۸۹

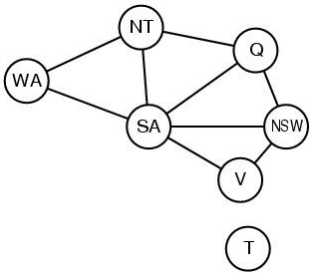
- سازگاری K با k=2، ناسازگاری را تشخیص می‌دهد.
- سازگاری K با k=3 نیز ناسازگاری را تشخیص خواهد داد (بررسی کنید: برای این منظور گره‌ها را به جای ۲تایی، ۳تایی با هم در نظر بگیرید).
  - بنابراین گزینه ۲ پاسخ است.

عقبگرد هوشمند

- الگوریتم عقبگرد وقتی با شکست مواجه شود، به سمت متغیر قبلی عقبگرد می‌کند (یک سطح) و مقدار دیگری برای آن در نظر می‌گیرد.
- در عقبگرد هوشمند، به جای اینکه یک مرحله به عقب برگردیم و شرایط را تکرار کنیم، آنقدر به عقب بر می‌گردیم، تا مشکل رفع شود (Backjumping).
  - مجموعه‌ای که می‌توانند باعث بروز مشکل شوند، مجموعه‌ی برخورد (Conflict Set) نام دارد.
  - مجموعه‌ی برخورد مربوط به یک متغیر شامل متغیرهایی است که قبلا مقدار گرفته و در گراف محدودیت به این متغیر خاص متصل هستند.

مثال: عقبگرد هوشمند

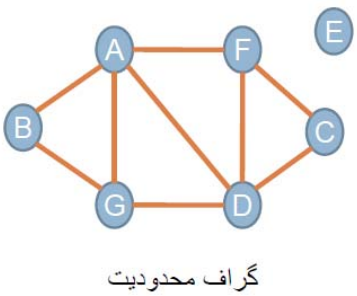
- اگر انتساب {Q=red, NSW=green, V=blue,T=red} انجام شود و نوبت به SA برسد، هیچ مقداری برای آن وجود ندارد.
  - الگوریتم عقبگرد معمولی به T باز می‌گردد و با تغییر مقدار آن دوباره راهش را ادامه می‌دهد، ولی تغییر مقدار این متغیر مشکل را حل نمی‌کند.
  - الگوریتم عقبگرد هوشمند، از مجموعه‌ی برخورد آخرین عضو را برای تغییر انتخاب می‌کند.
    - مجموعه برخورد در انتساب بالا {Q, NSW, V} است.



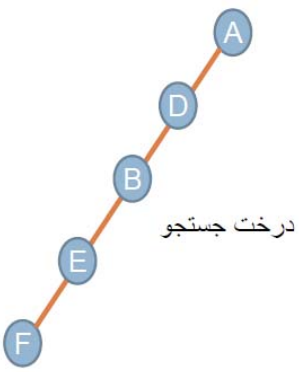
مهندسی ۸۸

- یک مسئله CSP دارای گراف محدودیت مقابل است. فرض کنید فضای حالت با یک جستجوی اول عمق پیمایش شود و با برخورد به یک شکست در مقداردهی اولیه به F در لحظه ای که درخت جستجوی آن نشان داده شده، نیاز به عقبگرد باشد. در عقبگرد براساس مجموعه تناقض (Conflict Set) به کدام گره باز خواهیم گشت؟

1. A
2. B
3. D
4. E



گراف محدودیت



درخت جستجو

مهندسی ۸۸

- گزینه ۳
  - از آنجا که متغیرهای E و B محدودیتی روی متغیر F اعمال نمی‌کند، بازگشت به E یا B و تغییر مقادیر آنها مقدار جدید معتبری برای F ایجاد نمی‌کند و مقدار متغیری باید تغییر کند که روی F محدودیت ایجاد کرده است.

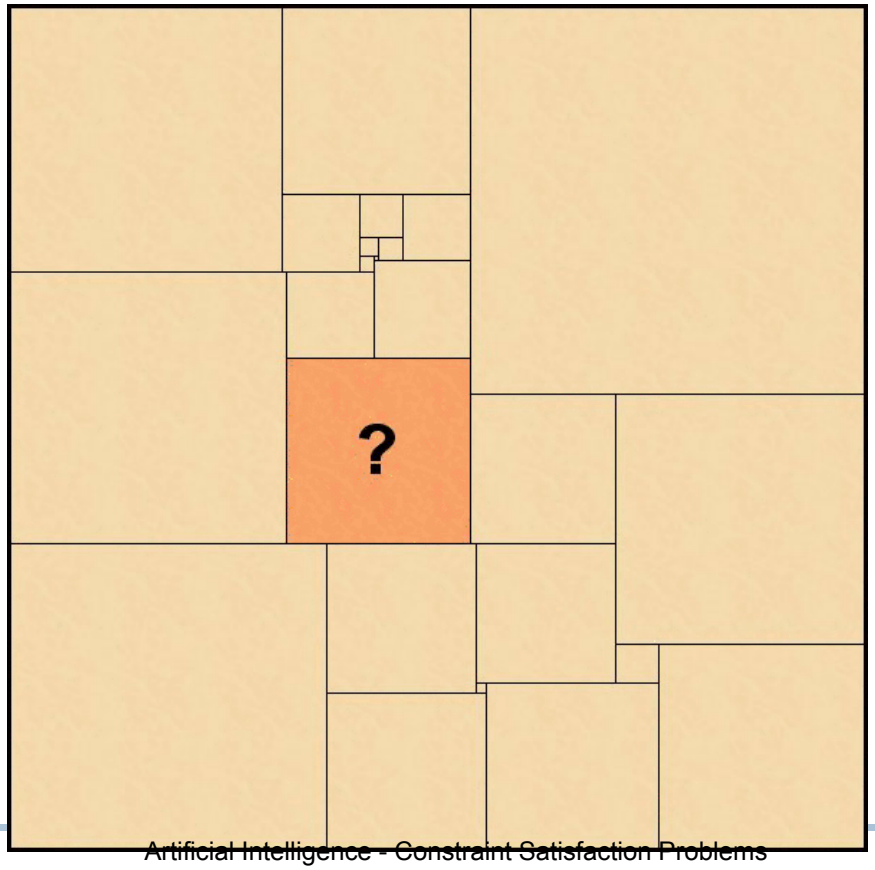
A crossword puzzle

```
    1   2   3   4   5
  +---+---+---+---+---+
1 | a |   | b |   | c |
  +---+---+---+---+---+
2 | # | # |   | # |   |
  +---+---+---+---+---+
3 | # | d |   | e |   |
  +---+---+---+---+---+
4 | f | # | g |   |   |
  +---+---+---+---+---+
5 | h |   |   |   |   |
  +---+---+---+---+---+
6 |   | # | # |   | # |
  +---+---+---+---+---+
```

```
Given the list of words:
        AFT      LASER
        ALE      LEE
        EEL      LINE
        HEEL     SAILS
        HIKE     SHEET
        HOSES    STEER
        KEEL     TIE
        KNOT

The places a,b,c,d,e,f,g,h in the crossword
puzzle correspond to the words that will start
at those locations.
```

The square below contains 24 smaller squares, each with a different integral size. Determine the length of the shaded square



The High IQ Exam

- From the High-IQ society entrance exam
  - Published in the Observer newspaper
  - Never been solved...
- Solved using a **constraint solver**
  - 45 minutes to **specify** as a CSP (Simon)
  - 1/100 second to solve (Sicstus Prolog)

High-IQ problem CSP

- Variables:
  - 25 lengths (Big square made of 24 small squares)
- Values:
  - Let the tiny square be of length 1
  - Others range up to about 200 (at a guess)
- Constraints: lengths have to add up
  - e.g. along the top row
- Solution: set of lengths for the squares
- Answer: length of the 17th largest square

سوال

- آیا جستجوهای زیر برای استفاده در یک مسئله ارضای محدودیت مناسب هستند؟
  - تپه نوردی
    - ممکن است مسأله حالت کامل نباشد و نتوان از الگوریتم های محلی استفاده کرد
  - اول عرض
    - مشکل عمده ای که وجود دارد فاکتور انشعاب است: در سطح اول nd (n تعداد متغیرها و d تعداد valueها)، در سطح بعدی (n-1)d (چون در سطح قبل یک انتساب صورت گرفته است) و... پیچیدگی زیاد است
  - عمیق ساز تکراری
    - زمانی استفاده می شود که هیچ اطلاعاتی در مورد عمق پاسخ نداشته باشیم، در صورتی که در مسائل CSP تعداد متغیرها محدود است.

فرمول‌بندی حالت کامل (Complete State Formulation)

- تپه نوردی عمدتا با وضعیت‌های کامل (وضعیت‌هایی که تمامی مقداری به تمامی متغیرها منتسب شده) کار می‌کند.
- در CSP مسیری که از طریق آن راه حل به دست می‌آید، اهمیت ندارد، بنابراین می‌توان از فرمول سازی حالت کامل استفاده کرد.
  - در این روش هر حالت یک انتساب کامل است که ممکن است محدودیت‌ها را ارضا کند یا نکند.
- الگوریتم های جستجوی محلی برای این روش خوب عمل می کنند.
  - در اینجا نیزهزینهٔ مسیر اهمیتی ندارد.

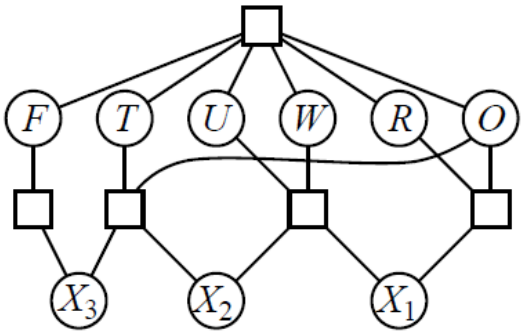
جستجوی محلی در مسائل ارضای محدودیت

- بسیاری از CSPها را بطور کارآمد حل می‌کنند
  - از فرموله‌سازی حالت کامل استفاده می‌کنند
  - حالت اولیه، مقداری را به هر متغیر نسبت می‌دهد
  - تابع جانشین، تغییر مقدار یک متغیر در هر زمان است
- انتخاب مقدار جدید برای یک متغیر
  - انتخاب مقداری که کمترین برخورد را با متغیرهای دیگر ایجاد کند
  - زمان اجرای کمترین برخورد، مستقل از اندازه مسأله است
- مزیت اصلی جستجوی محلی
  - در صورت تغییر مسأله تنظیمات را به صورت Online انجام می‌دهد
  - این موضوع در مسایل زمانبندی بسیار مهم است

Criptarithmetic Problem

- یک مسئله با محدودیت مرتبه بالاتر و استفاده از متغیر کمکی

$$\begin{array}{r} T\ W\ O \\ +\ T\ W\ O \\ \hline F\ O\ U\ R \end{array}$$



Criptarithmetic Problem

- Variables:
  - F T U W R O X1 X2 X3
- Domains:
  - {0,1,2,3,4,5,6,7,8,9}
- Constraints: Alldiff (F,T,U,W,R,O)
  - O + O = R + 10 * X1
  - W + W + X1 = U + 10 * X2
  - T + T + X2 = O + 10 * X3
  - X3 = F
  - $(O \neq T \neq W \neq R \neq U \neq F)$
  - (O, T, W, R, U, F) $\in N\ and\ 0 \leq x \leq 9$
  - $\forall i\ (Xi) \rightarrow (X \in N) and\ 0 \leq X \leq 1$

Criptarithmetic Problem

- تابع All Different مشخص می‌کند که مقادیر انتخاب شده برای متغیرها بایستی همگی با یکدیگر متفاوت باشند. این محدودیت طبق تعریف مسئله معمای ریاضیات رمزی اضافه شده. یعنی صورت مسئله از ما چنین خواسته است که مقادیر انتخابی بایستی همه با هم متفاوت باشند لذا در طراحی محدودیت‌ها این موضوع را اعمال می‌کنیم.
- علاوه بر این مقادیر انتخابی برای متغیرها بایستی مقادیر بین صفر تا ۹ و فقط در حوزه دامنه اعداد طبیعی باشند. مثلا برای متغیر R نمی‌توانیم عدد ۱۲ را انتخاب کنیم. این موضوع نیز از صورت مسئله استخراج شده که طبق تعریف مسئله معمای ریاضیات رمزی مقادیر متغیرها نمی‌توانند اعداد اعشاری یا غیر از یک رقمی باشند.
- محدودیتی روی مقادیر نقلی اعمال شده است که دامنه آن‌ها محدود به دو حالت صفر و یک است. زیرا در بهترین و بدترین شرایط جمع دو عدد یک رقمی فقط یکی از این دو عدد به عنوان نقلی انتخاب می‌شوند. لذا این محدودیت روی مقادیر نقلی اعمال شده است.

Criptarithmetic Problem

- محدودیت‌های مرتبه بالاتر می‌تواند با ابر گراف محدودیت HyperGraph نمایش داده شوند. محدودیت ALLDiff را می‌توان به محدودیت‌های دودویی $F \neq T$ و $F \neq U$ تبدیل کرد. اگر متغیرهای کمکی کافی در نظر گرفته شود هر محدودیت مرتبه بالاتر با دامنه متناهی می‌تواند به مجموعه‌ای از محدودیت‌های دودویی کاهش یابد.