Zadání semestrální práce z předmětu Evoluční optimalizační algoritmy a nabídka témat.

Zadání a podmínky vypracování SP

- **Zadání**
  1. Implementace lokálního prohledávacího algoritmu
  2. Implementace jednoduchého evolučního algoritmu
  3. Implementace specializovaného EA nebo memetického algoritmu
- **Důležité body návrhu optimalizačních algoritmů**
  - Reprezentace řešení
  - Variační operátory u lokálního prohledávání
  - Operátory křížení a mutace u EA
  - Ohodnocovací funkce
- **Zpracování**
  - Fungující program
  - Závěrečná zpráva
  - Prezentace

Vše musí být odevzdáno ve stanovených termínech!
Pozdní odevzdání bude penalizováno 4 bodovou srážkou za každý započatý týden.

Zpracování SP

- Fungující program
  - Fungující kód pro všechny řešené úlohy.
  - GUI není vyžadováno (ale může být oceněno bonusovými body).
- Závěrečná zpráva
  - musí obsahovat tabulky a grafy se statistickým zhodnocením provedených experimentů na **zadaných testovacích datech**;
  - musí obsahovat **grafy s průběhem mediánu nejlepší fitness v závislosti na počtu ohodnocení**;
  - musí obsahovat grafy s průběhem **mediánu nejlepší hodnoty jednotné ohodnocovací funkce v závislosti na počtu ohodnocení**.
- Prezentace
  - **Společná část** - skupina studentů, řešící stejnou úlohu, vypracuje společně úvodní slajdy představující úlohu a závěrečné slajdy s prezentací výsledků a vzájemným porovnáním přístupů jednotlivých studentů.
  - **Individuální část** – slajdy stručně a srozumitelně popisující zvolenou reprezentaci, operátory a ohodnocovací funkci.

1. Japanese puzzle - nonogram

- **Popis problému**: Nonogram se skládá se ze tří částí:
  - mřížka obdélníkového tvaru *M*x*N*, do které se má vyplnit obrázek z plných a prázdných políček,
  - dvě legendy (levá a horní). Každému řádku obrázku odpovídá řádek levé legendy, sloupci obrázku odpovídá sloupec horní legendy.

    V řádcích/sloupcích legend jsou uvedeny seznamy celých čísel. Každé číslo odpovídá souvislému bloku plných políček dané délky. Pořadí čísel v legendě určuje pořadí bloků v obrázku.

    Mezi dvěma sousedními bloky v obrázku musí být alespoň jedno prázdné políčko. Na začátku a na konci každého řádku a sloupce se může, ale nemusí, vyskytovat libovolný počet prázdných políček.

  Cílem je vyplnit mřížku plnými políčky tak, aby výsledný obrázek přesně odpovídal všem řádkům a sloupcům legendy.

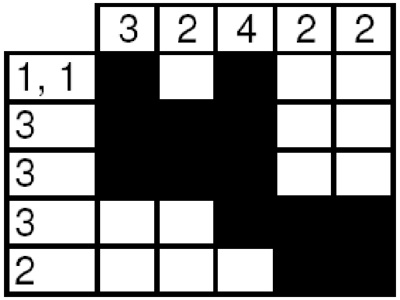
	3	2	4	2	2
1, 1					
3					
3					
3					
2					

http://www.puzzle-nonograms.com/

1. Japanese puzzle - nonogram

- Popis úlohy: Je dána mřížka *MxN* a u každého řádku a sloupce jsou posloupnosti čísel. Ty udávají jak velké souvislé bloky plných políček a v jakém pořadí se v daném sloupci/řádku očekávají.
  Cílem je nalézt konkrétní vyplnění plných políček v mřížce tak, aby všechna omezení byla splněna.

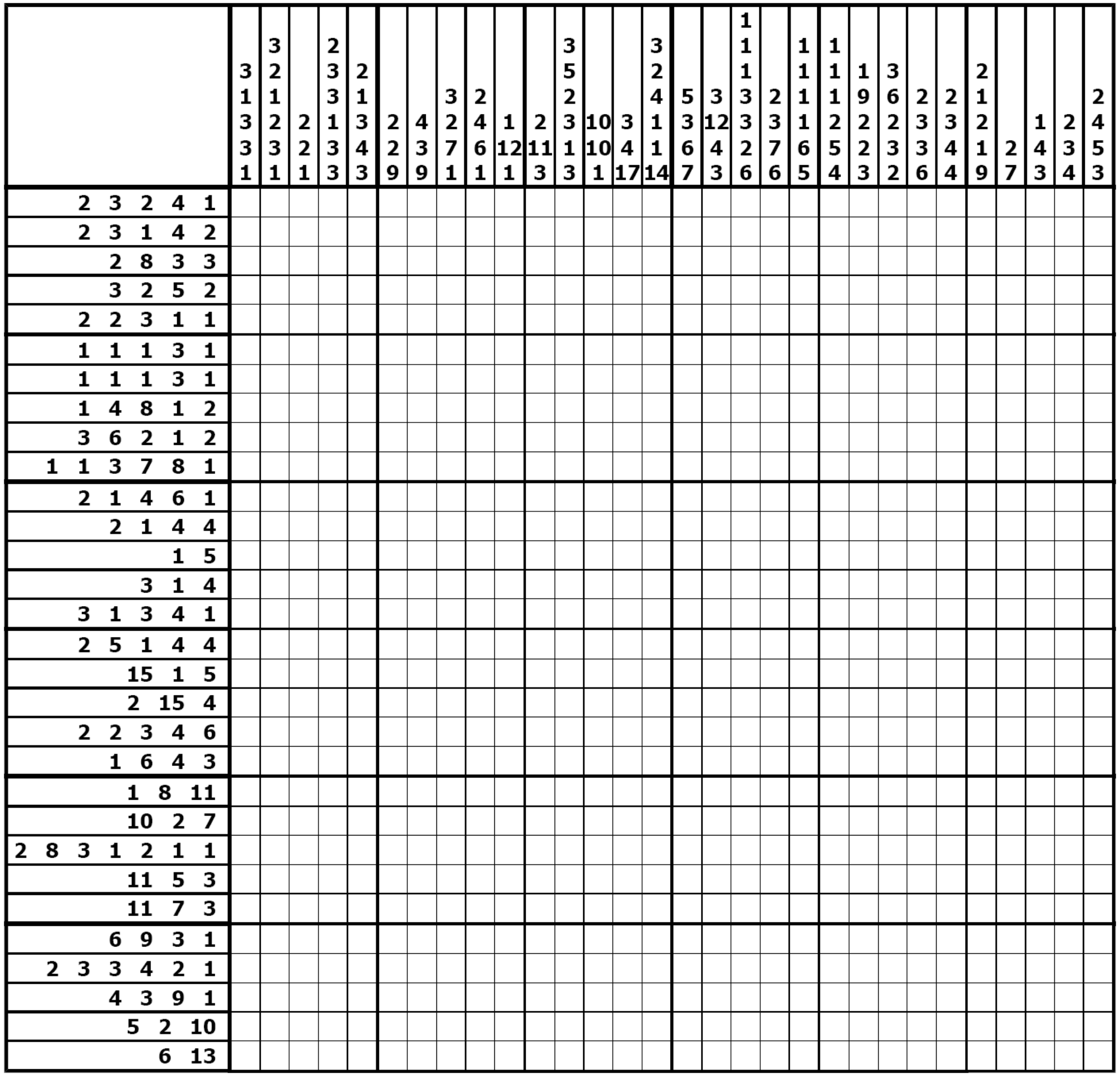
  Proč na to programovat evoluční algoritmus?



http://www.puzzle-nonograms.com/

1. Japanese puzzle - nonogram

Proto ...

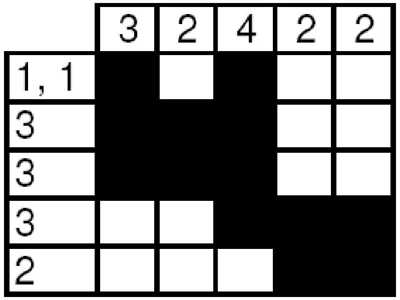


																	1												
	3		2								3			3			1		1	1									
3	2		3	2							5			2			1		1	1	1	3			2				
1	1		3	1			3	2			2			4	5	3	3	2	1	1	9	6	2	2	1				2
3	2	2	1	3	2	4	2	4	1	2	3	10	3	1	3	12	3	3	1	2	2	2	3	3	2		1	2	4
3	3	2	3	4	2	3	7	6	12	11	1	10	4	1	6	4	2	7	6	5	2	3	3	4	1	2	4	3	5
1	1	1	3	3	9	9	1	1	1	3	3	1	17	14	7	3	6	6	5	4	3	2	6	4	9	7	3	4	3

2 3 2 4 1
2 3 1 4 2
2 8 3 3
3 2 5 2
2 2 3 1 1
1 1 1 3 1
1 1 1 3 1
1 4 8 1 2
3 6 2 1 2
1 1 3 7 8 1
2 1 4 6 1
2 1 4 4
1 5
3 1 4
3 1 3 4 1
2 5 1 4 4
15 1 5
2 15 4
2 2 3 4 6
1 6 4 3
1 8 11
10 2 7
2 8 3 1 2 1 1
11 5 3
11 7 3
6 9 3 1
2 3 3 4 2 1
4 3 9 1
5 2 10
6 13

http://www.puzzle-nonograms.com/

1. Japanese puzzle - nonogram



- **Reprezentace:**
  - Binární vektor nebo binární matice $\{0, 1\}^{M+N}$
  - Sloupcové nebo řádkové bloky.
- **Jednotná ohodnocovací funkce**: Považujme řádkové a sloupcové komponenty legendy za řetězce celých čísel. Stejně tak reprezentace aktuálního stavu řádku a sloupce považujme za řetězce celých čísel. Potom míru shody mezi legendou a daným stavem na řádku/sloupci spočítáme jako podobnost dvou řetězců podle **Needleman-Wunchova algoritmu**.

  Výsledná hodnota shody legendy a konkrétního řádku/sloupce matice je součtem rozdílů hodnot přes všechny dvojice čísel na souhlasných pozicích a penalizací za vložené mezery.

  Celková kvalita řešení se počítá jako součet příspěvků spočítaných **přes všechny řádky a sloupce matice.**

1. Japanese puzzle - nonogram

- **Příklad** výpočtu (ne)shody

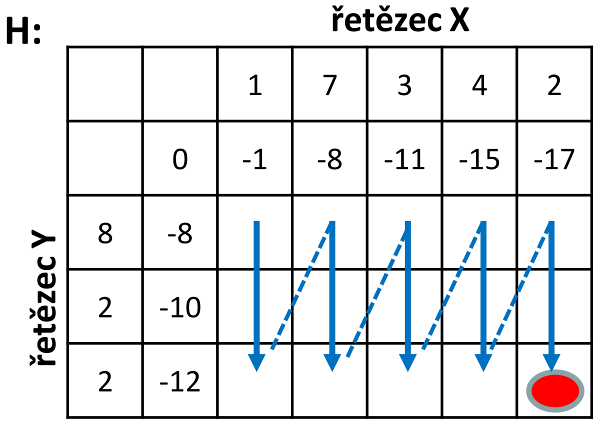
  legendy (řetězec X): 1 7 3 4 2

  a

  řádku matice (řetězec Y): 8 2 2

  pomocí Needleman-Wunchova algoritmu.

  Výpočet spočívá ve vyplnění tabulky shody H, viz obrázek, kde sloupxce reprezentují znaky řetězce X a řádky odpovídají znakům řetězce Y. Začíná se z levého horního políčka, které má hodnotu 0 a vyplňování pokračuje po sloupcích zleva doprava a ve sloupci shora dolů. Hodnota v pravém dolním políčku udává výslednou hodnotu neshody řetězců X a Y.



		1	7	3	4	2
	0	-1	-8	-11	-15	-17
8	-8					
2	-10					
2	-12					●

  Parametry Needleman-Wunchova algoritmu jsou:

  - **penalizace za mezeru**: -*K*, kde *K* je hodnota znaku, který je na stejné pozici s mezerou.

    Konkrétně, pokud uvažujeme přechod H(*i*-1, *j*) → H(*i*, *j*), tedy přechod dolů, tak za tuto vloženou mezeru je penalta rovna záporné hodnotě znaku na řádku *i*. Pokud uvažujeme přechod H(*i*, *j*-1) → H(*i*, *j*), tedy přechod doprava, tak za takto vloženou mezeru je penalizace rovna záporné hodnotě znaku ve sloupci *j*.
  - **Ohodnocení (ne)shody stejnolehlých znaků**: $-|X_j - Y_i|$, kde X_j a Y_i jsou čísla na stejnolehlých pozicích.

1. Japanese puzzle - nonogram

- **Příklad** výpočtu shody

legendy (řetězec X): 1 7 3 4 2

a

řádku matice (řetězec Y): 8 2 2

pomocí Needleman-Wunchova algoritmu.

řetězec X

řetězec Y

		1	7	3	4	2
	0	**-1**	-8	-11	-15	-17
8	-8	-7	**-2**	**-5**	-9	-11
2	-10	-9	-4	-3	**-7**	-9
2	-12	-11	-6	-5	-5	**-7**

Postup: Vyplň tabulku tak, že hodnota každého políčka se získá podle následujícího pravidla

$$\mathrm{H}(i, j) = \max(\mathrm{H}(i-1, j)-K_i, \mathrm{H}(i, j-1)-K_j, \mathrm{H}(i-1, j-1) + \mathit{neshoda}(i, j)),$$

kde K_i resp. K_j je hodnota i-tého znaku řetězce Y resp. hodnota j-tého znaku řetězce X.

$\mathit{neshoda}(i, j) = -(|X_j - Y_i|)$.

Výsledná hodnota je -7. Tomu odpovídá několik řešení, například

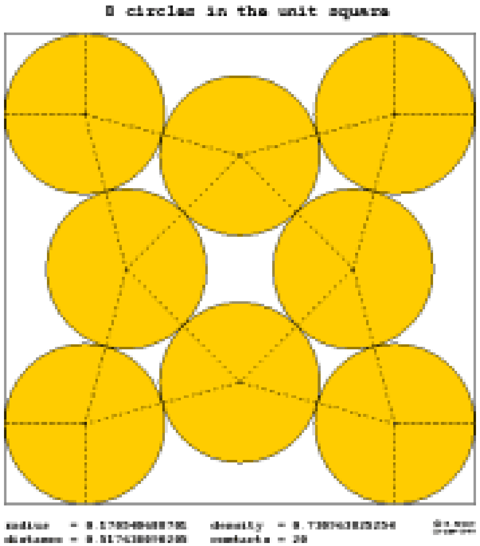
legenda (řetězec X): 1 7 3 4 2
řádku matice (řetězec Y): - 8 - 2 2

nebo

legenda (řetězec X): 1 7 3 4 2
řádku matice (řetězec Y): - 8 2 2 -

2. Kruhy ve čtverci

- **Popis problému**: Je dána čtvercová plocha o straně délky 1.
  Cílem je umístit na tuto plochu *N* stejně velkých kruhů s maximálním poloměrem *r* tak, aby se žádné dva nepřekrývaly a žádný nevyčníval vně této plochy.
  Vstup: Hodnota parametru *N*.
  Výstup: Poloměr *r* a souřadnice středů kruhů.
- **Reprezentace**: Seznam souřadnic středů kruhů, tedy seznam dvojic [x_i, y_i] pro *i*=1...*N*; poloměr *r* se z toho dopočítá.
- **Jednotná ohodnocovací funkce**: Kvalita řešení se počítá jako největší možný poloměr kruhů pro danou konfiguraci středů.
  Tato funkce je maximalizována.



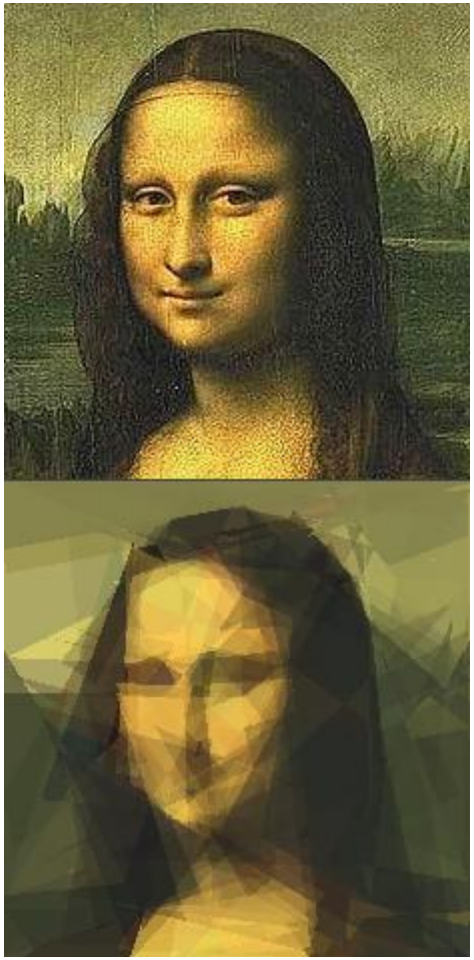
www.packomania.com

3. Ztrátová komprese obrázku

- **Popis úlohy**: Uvažujme obrázek v bitmapovém formátu – může být černobílý i barevný.
  Cílem je pro zvolenou reprezentaci, tj. překrývající se polo- nebo neprůhledné polygony, elipsy, nebo kružnice, nalézt komprimovanou formu obrazu, která má minimální odchylku od původního obrazu.

  Vstup: Počet a typ základních primitiv.

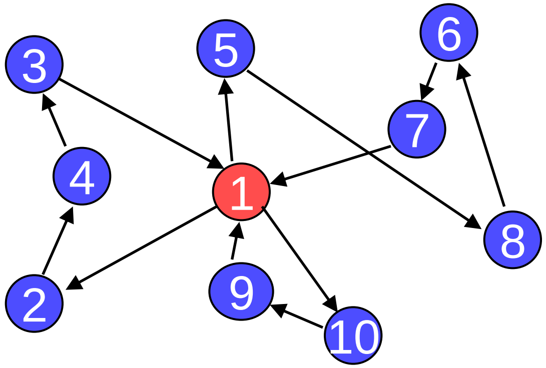
  Výstup: Transformovaný obraz.
- **Jednotná ohodnocovací funkce**: Kvalita komprese se počítá jako celková odchylka přes všechny pixely a složky jasu (RGB).
- **Možné typy reprezentace**
  - Poloprůhledné n-úhelníky, kruhy nebo elipsy, které se překrývají – intenzita jasu se sčítá.
    Neprůhledné kruhy nebo elipsy, které se překrývají.
  - Hladká funkce intenzity pro každou složku RGB.
    Funkce $I_R(x,y)$, $I_G(x,y)$, $I_B(x,y)$.
  - Oblasti oddělené pomocí Voronoiova diagramu.



Sean Luke. 2009. *Essentials of Metaheuristics*.
http://cs.gmu.edu/~sean/book/metaheuristics/

4. TSP s více cestujícími

- **Popis problému**: Na vstupu je úplný neorientovaný graf o *N* uzlech.
  Cílem je nalézt takovou množinu cest pro *M* cestujících, které všechny vychází z počátečního uzlu (depotu) a zase v něm končí. Všechny uzly (kromě depotu) jsou navštíveny právě jednou a délka nejdelší cesty je minimální. Žádná z cest nesmí mít nulovou délku.



- **Reprezentace**: [permutace měst][break-pointy]

  příklad: [2-4-3-5-8-6-7-10-9][3,7]

- **Jednotná ohodnocovací funkce**: Kvalita řešení je určena jako délka nejdelší cesty z *M* cest.

5. Warehouse Location

- **Popis problému**: Distribuční firma obsluhuje skupinu geograficky rozptýlených zákazníků z nichž každý požaduje jisté množství odebíraného zboží. Firma má k dispozici několik možných lokací pro sklady svého zboží, každý sklad má svoji kapacitu.
  Cílem je přiřadit zákazníky ke skladům tak, aby bylo dosaženo maximálně efektivního obsloužení všech zákazníků.

  Vstup:
  - *N* skladů, každý sklad *w* má svoji kapacitu cap_w a zřizovací cenu s_w,
  - *M* zákazníků, každý zákazník *c* požaduje jiné množství zboží d_c,
  - Pro každou dvojici <*c*,*w*> je definována cena, t_{cw}, za doručení zboží ze skladu *w* k zákazníkovi *c*.

  Výstup: Přiřazení zákazníků ke skladům tak, aby byla minimalizována **jednotná ohodnocovací funkce**

$$f(x) = \sum_{w \in N} \left(\left(|a_w| > 0 \right) s_w + \sum_{c \in a_w} t_{cw} \right)$$

  za podmínek $\sum_{c \in a_w} d_c \leq cap_w$ a $\sum_{w \in N} \left(c \in a_w \right) = 1$ pro všechny $w \in N$ a $c \in M$

  kde a_w je množina zákazníků přiřazených ke skladu *w*.

5. Warehouse Location

- **Reprezentace**:
  - pole o velikosti *M*, kde *i*-tá hodnota reprezentuje číslo skladu přiřazené *i*-tému zákazníkovi,
  - matice $A[M \text{x} N]$, kde $A_{cw} \in \{0, 1\}$
    A_{cw} = 1, když je zákazník *c* přiřazen ke skladu *w*
    = 0, jinak.

6. CD kompilace

- **Popis problému**: Rocková skupina XY chce na sklonku kariéry vydat soubornou kompilaci všech svých vypalovaček. Problém je, jak skladby optimálně rozvrhnout na co nejmenší počet CD nosičů stejné kapacity *C*.

  Cílem je vměstnat všechny skladby na co nejmenší počet disků.

  Vstup: *N* skladeb, každá skladba má svoji délku v sekundách.
  Jednotná kapacita disků *C*.

  Výstup: Hodnota průměru kvadrátů zaplněnosti použitých disků.

- **Reprezentace**:
  - Seznam lineárních řetězců celých čísel, kde každý řetězec reprezentuje jeden disk a čísla v řetězci udávají identifikátory skladeb.
  - Dva seznamy – seznam (permutace) skladeb, seznam rozdělujících bodů (pozice v rámci seznamu skladeb, které vymezují jednotlivé disky).
    [permutace skladeb][break-pointy]

6. CD kompilace

- **Jednotná ohodnocovací funkce:** Celková zaplněnost disků řešení *s* s *M* disky je

$$f(s) = \frac{\sum_{i=1}^{M} (z_i / C)^2}{M}$$

kde z_i je zaplněnost i-tého CD.

Tato funkce je maximalizována.

7. Hledání nejkratší společné supersekvence

- **Popis problému**: Je dána množina řetezců znaků dané abecedy. Cílem je nalézt takovou posloupnost znaků dané abecedy (supersekvenci), že všechny původní řetezce jsou v ní zcela obsažené. Řetězec *r* je obsažen v supersekvenci *S* právě tehdy když všechny znaky řetězce *r* jsou přítomny v supersekvenci *S* a to v pořadí, v jakém se vyskytují v *r*.

  Vstup: Abeceda *A*, ze které jsou tvořeny řetězce.

  N řetězců (ne nutně stejné délky).

  Výstup: Supersekvence *S* splňující výše uvedenou vlastnost.

- **Reprezentace**: Lineární řetězec znaků dané abecedy.

Př.:

```
             s1:  ca ag cca  cc ta     cat  c a
             s2:  c gag ccat ccgtaaa g  tt  g
             s3:   aga acc tgc  taaatgc t a ga
------------------------------------------------------
Supersequence S:  cagagaccatgccgtaaatgcattacga
```

7. Hledání nejkratší společné supersekvence

- **Jednotná ohodnocovací funkce:** Kvalita supersekvence S je počítáná podle ohodnocovací funkce

  $$f(S) = C(S) + L(S),$$

  kde $C(S)$ je celkový počet znaků, které S pokrývá a $L(S)$ je příspěvek za délku supersekvence počítaný jako

  $$L(S) = (SumL - Length(S)) / SumL.$$

  SumL je součet všech znaků ve vstupních řetězcích.

  Tato funkce je maximalizována.

8. Sestavování žebříčku ATP

- **Popis problému**: Máme bilanci výsledků vzájemných zápasů tenistů na okruhu ATP. Data jsou uložena v matici ***B***, kde hodnotu na pozici [*i*, *j*] může vyjadřovat absolutní nebo relativní bilanci mezi hráči *i* a *j*:
  - *I.* $b_{ij} = n$; hráč *i* v sezoně *n*-krát zvítězil nad hráčem *j*
  - *II.* $b_{ij} = 1$; hráč *i* má pozitivní bilanci s hráčem *j*
    $b_{ij} = 0$; hráč *i* má negativní bilanci s hráčem *j*
- **Reprezentace**: Lineární sekvence (permutace) hráčů.
- **Jednotná ohodnocovací funkce.** Kvalita daného žebříčku hráčů (permutace hráčů, π) se počítá pomocí následující funkce

$$f(\pi) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} B_{\pi(i)\pi(j)}$$

Tato funkce je maximalizována.

9. Rozděl a panuj!

- **Popis problému**: Jste správci rozlohou velké oblasti s množstvím měst s hustou sítí silnic. Každodenní správcovské povinnosti už vás ale nebaví a chtěli byste práci delegovat na několik svých zástupců. Je proto třeba rozdělit celé panství na několik stejně velkých regionů, každý z nich bude řídit jeden zástupce.
  Cílem je najít rozdělení původní oblasti na maximálně autonomní regiony tj. s minimálním propojením mezi regiony.

  Vstup: Graf $G(V, E)$, kde V je množina měst $\{v_1,\dots,v_M\}$ a E je množina silnic spojující města.

  Výstup: Rozdělení množiny V na vzájemně disjunktní podmnožiny $V_1, \ldots, V_n$,

  kde $\bigcup_{i=1}^{n} V_i = V$ a $\left\| V_i \right| - \left| V_j \right\| \leq 1$ pro všechny $i \neq j$.

- **Reprezentace**: Pole velikosti M, kde i-tá pozice obsahuje číslo regionu, do kterého i-té město patří.

- **Jednotná ohodnocovací funkce:** Počet cest mezi městy z jiného regionu.

$$f(x) = \sum_{l \in V_i, k \in V_j, i \neq j} e_{lk}$$

  Tato funkce je minimalizována.