Desarrollo de algoritmos inteligentes en problemas de optimización: Estudio de un caso práctico en sistemas territoriales

Roger Z. Ríos

Programa de Posgrado en Ing. de Sistemas
Universidad Autónoma de Nuevo León

http://pisis.fime.uanl.mx/

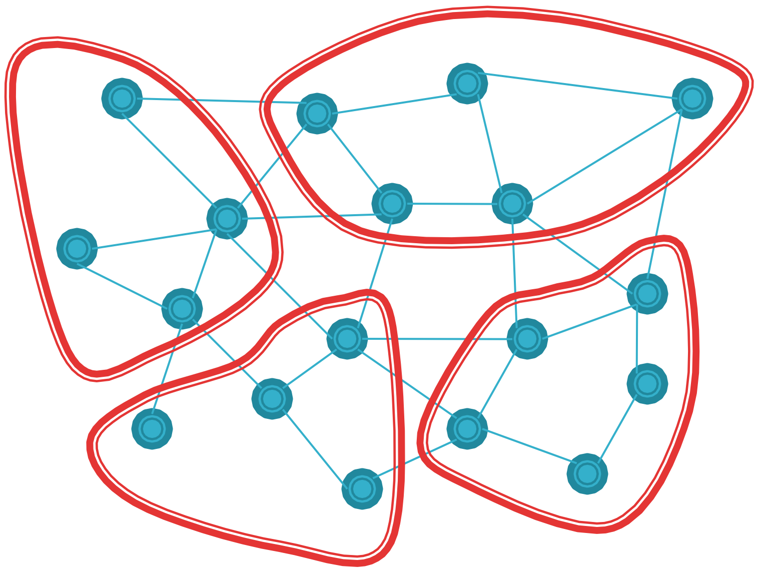
Ciclo de Conferencias 2010
F C F M – U A N L

Cd. Universitaria 09 Octubre 2010

Agenda

- Territory design/districting background
- Examples:
  - Commercial territory design
- Models & Algorithms:
  - Mathematical programming
  - metaheuristics
- Wrap-up

Background: Territory design/districting



Data

- Node → Basic unit (BU)
- BU location (coordinates)
- BU activity
- p territories

Generic problem specification

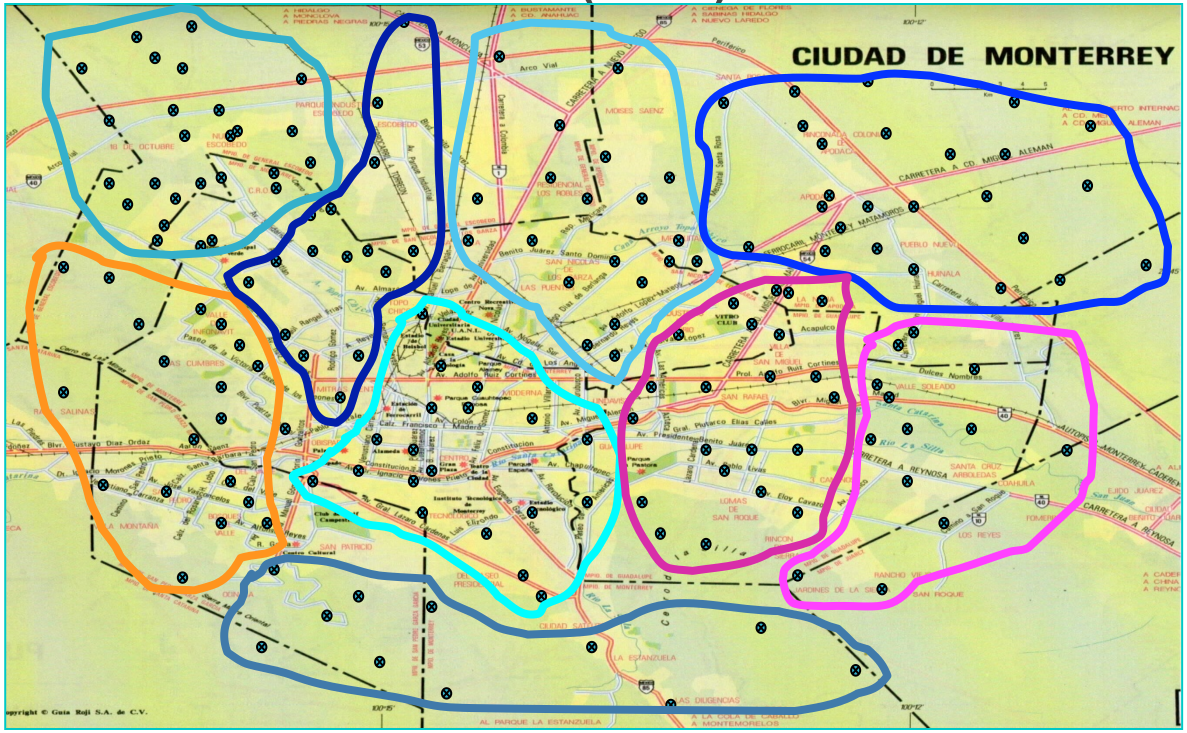
The objective is to find a p-partition of a given set of basic units (BUs) that optimizes a performance measure subject to specified planning criteria.

Background: Applications

- Political districting
  - Hess et al. (1965)
  - Garfinkel and Nemhauser (1970)
  - Hojati (1988)
  - Ricca and Simeone (1997)
  - Mehrotra et al. (1998)
  - Bozkaya, Erkut, and Laporte (2003)
- Sales territory design
  - Hess and Samuels (1971)
  - Shanker et al. (1975)
  - Zoltners (1979)
  - Zoltners and Sinha (1983)
  - Fleischmann and Paraschis (1983)
  - Drexl and Hasse (1999)
- Service territory design
  - Easingwood (1973)
  - Marlin (1981)
  - Blais et al. (2003)
- School districting
  - Palermo et al. (1977)
  - Ferland and Guénette (1990)
- Waste recollection
  - Muyldermans et al. (2002)
  - Hanafi et al. (1999)
  - Fernández et al. (forthcoming)
- Commercial territory design
  - Ríos-Mercado and Fernández (2009)
  - Caballero-Hernández and Ríos-Mercado (2008)
  - Segura-Ramiro et al. (2008)

Example B: Commercial territory design

Ríos-Mercado & Fernández (2009)

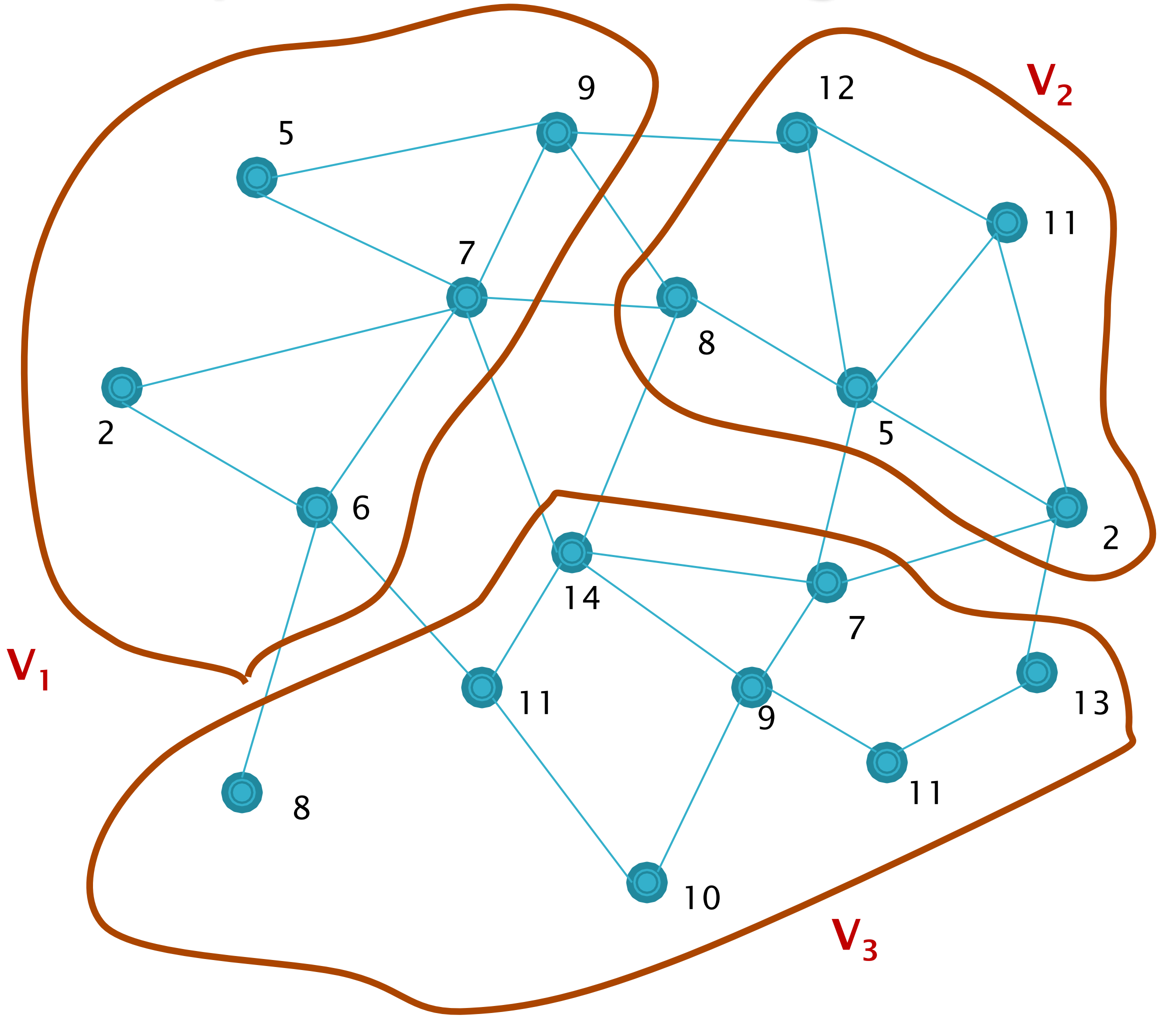


- w(j,a) = weight of activity a in unit j
- d(i,j) = Euclidean distance between units i and j

Example B: Commercial territory design (cont'd)

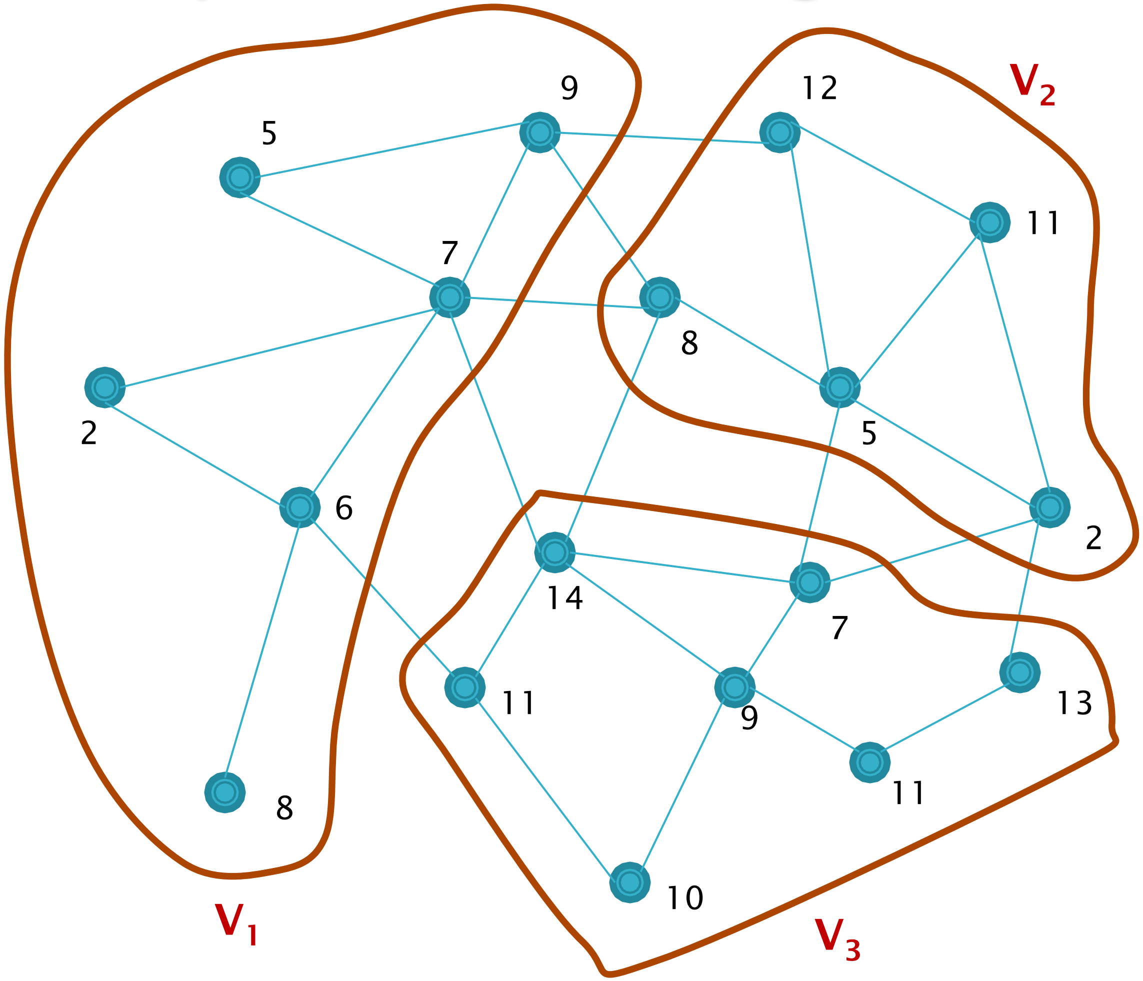
- **Objective:** Partition territory into commercial districts
- Fixed number of districts (*p*)
- Territory balance (multiple node activities)
  - Demand
  - Number of customers
- Contiguity
- Compactness

Example B: Illustrating connectivity



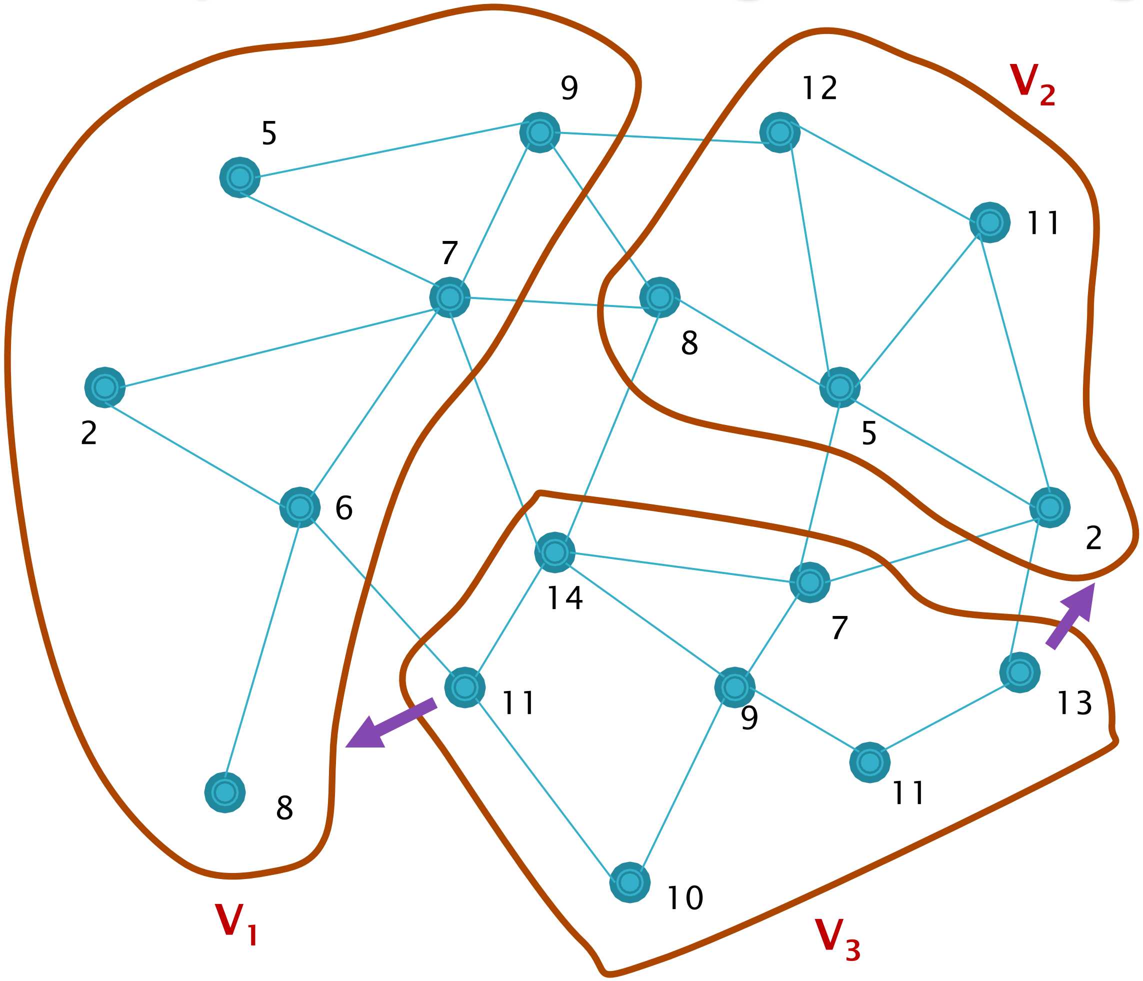
Unconnected territories

Example B: Illustrating connectivity



Connected territories

Example B: Modeling balancing



Modeling balance

Goal for average demand

$\mu := \left(\sum w_j \right) / p$

$\mu := 150 / 3 = 50$

where

w_j := demand at node j

p := number of territories (= 3)

τ := Tolerance parameter (= 0.10)

Territory size w(V)

Territory is balanced if

$(1-\tau)\mu \leq w(V_j) \leq (1+\tau)\mu$

$45 \leq w(V_j) \leq 55$

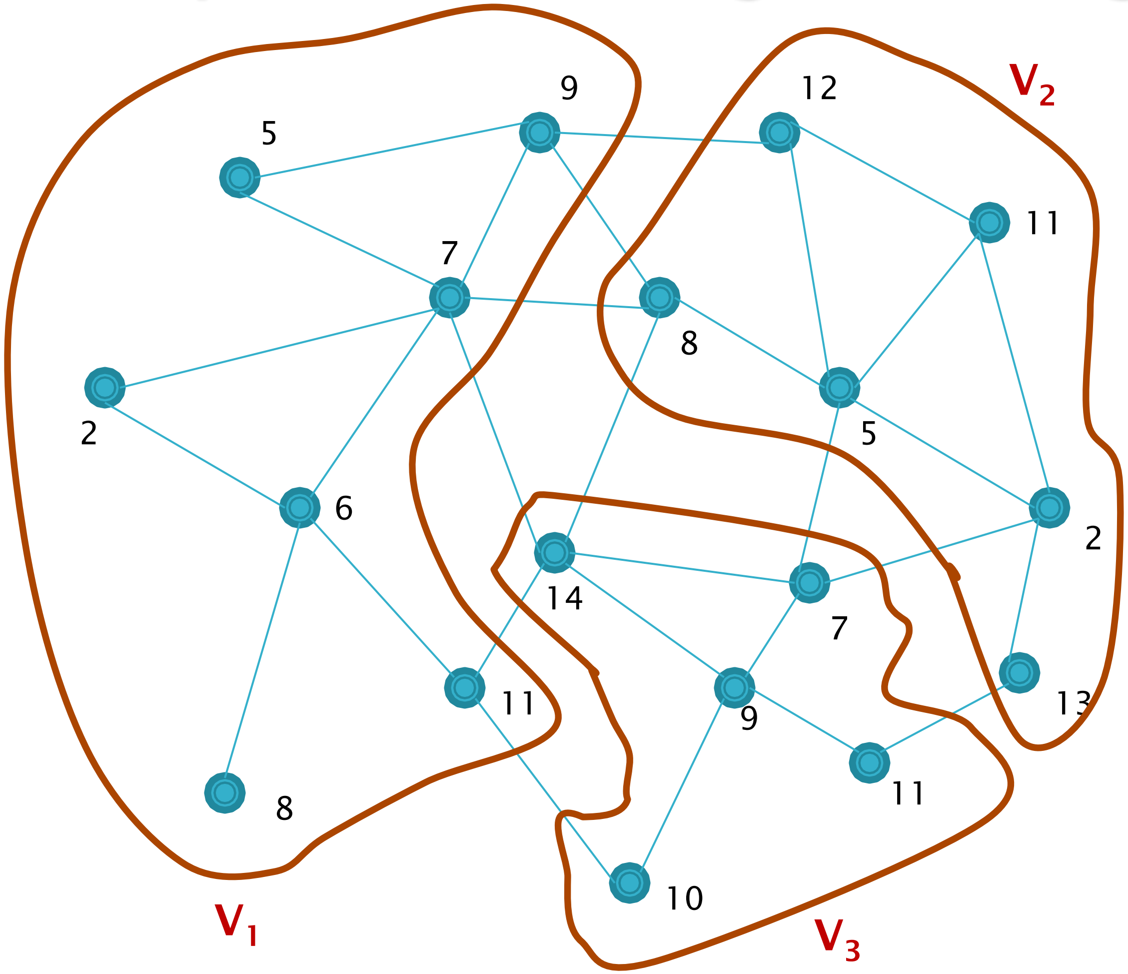
In this example:

$w(V_1) = 37$ [Unbalanced]

$w(V_2) = 38$ [Unbalanced]

$w(V_3) = 75$ [Unbalanced]

Example B: Modeling balancing



Modeling balance

Goal for average demand

$\mu := \left(\sum w_j \right) / p$

$\mu := 150 / 3 = 50$

where

w_j := demand at node j

p := number of territories (p = 3)

τ := Tolerance parameter (= 0.10)

Territory size w(V)

Territory is balanced if

$(1-\tau)\mu \leq w(V_j) \leq (1+\tau)\mu$

$45 \leq w(V_j) \leq 55$

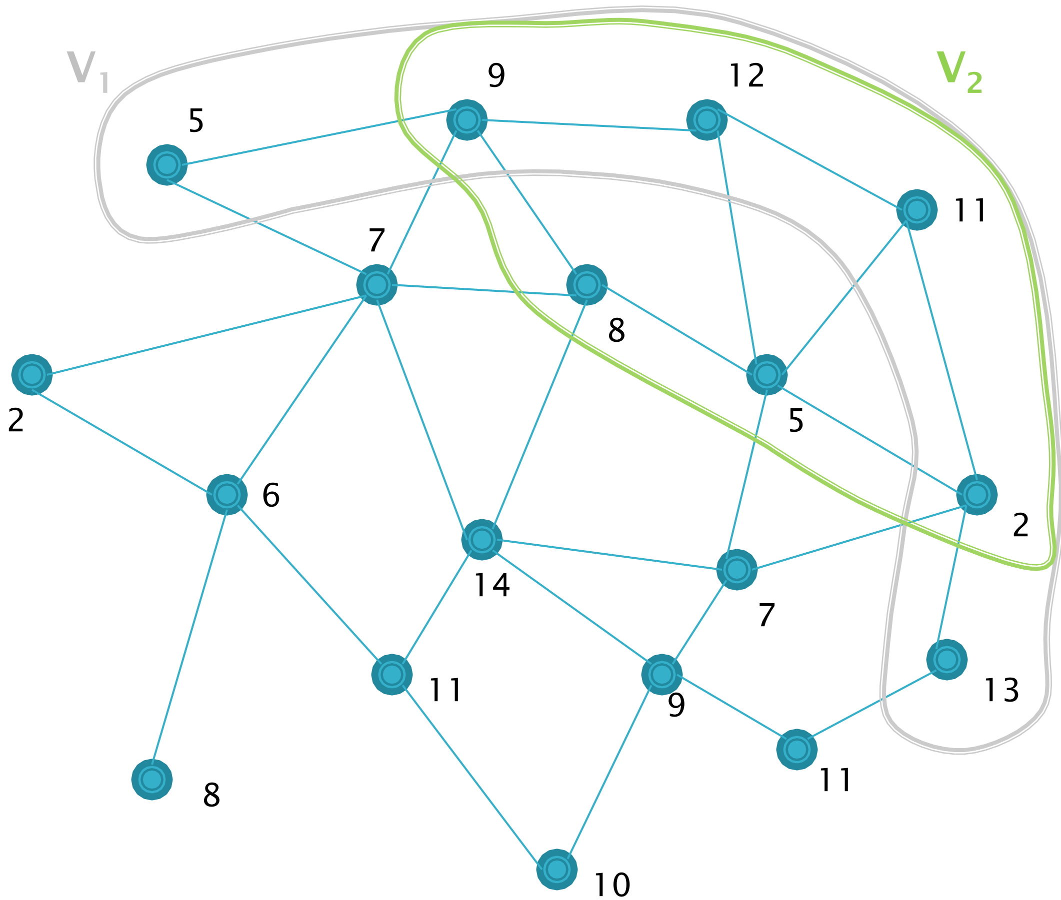
In this example:

$w(V_1) =$ 48 [Balanced]

$w(V_2) =$ 51 [Balanced]

$w(V_3) =$ 51 [Balanced]

Example B: Illustrating Compactness



V_2 is "more compact" than V_1

Modeling compacity

d_{ij} := Distance from node i to j

Territory centers

c(k) := center of territory k

$d(V_k)$:= dispersion function

$d(V_k)$ = sum { $d_{c(k),j}$: j in V_k }

$d(V_k)$ = max { $d_{c(k),j}$: j in V_k }

Objective (dispersion)

Minimize $\sum_{k=1}^{p} d(V_k)$

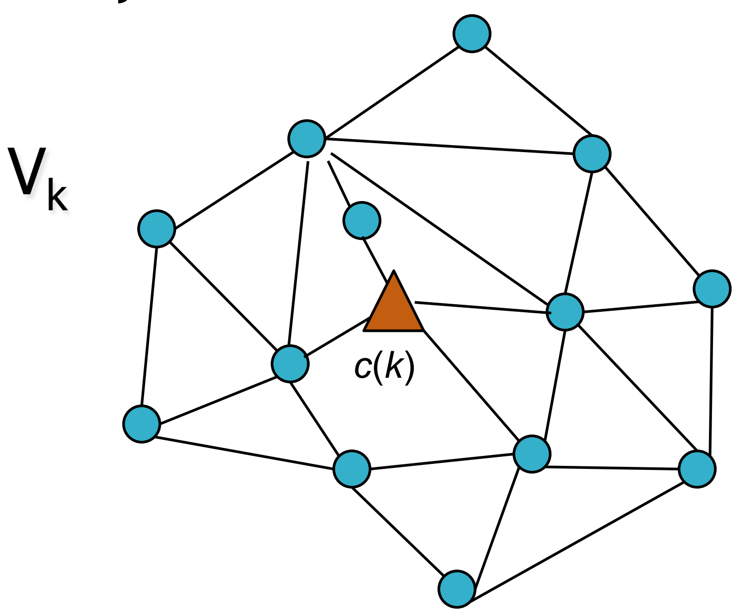
Minimize   max $d(V_k)$

Example B: Motivation of decision variables

Modeling compactness:
Introduce territory "centers"

$$c(k) = \arg\min_{i \in V_k} \left\{ \sum_{j \in V_k} d_{ij} \right\}$$

$$c(k) = \arg\min_{i \in V_k} \left\{ \max_{j \in V_k} \{ d_{ij} \} \right\}$$



$$x_{ij} = \begin{cases} 1 & \text{if unit } j \text{ is assigned to territory with center in } i \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ii} = \begin{cases} 1 & \text{if center is placed at node } i \\ 0 & \text{otherwise} \end{cases}$$

Basic model: Description

Constraints

- Each BU (node) must belong to a unique territory
- For each territory, the activity value is the sum of the activity value of its BUs
- Territories must be "balanced" with respect to the activity measure
- Number of territories is *p*.

Objective

- Minimize territory dispersion (maximize compactness)

Basic model: Parameters

- $G = (V,E)$ Problem graph
- $V = \{ 1, 2, ..., n \}$ Set of BUs (nodes)
- E := Set of edges; (i, j) exists if i and j are adjacent blocks
- p := Number of territories
- w_j:= Demand in node $j; j \in V$
- μ := Target of node activity given by $\left(\sum_{j \in V} w_j \right) / p$
- τ := Tolerance parameter regarding μ
- d_{ij}:= Euclidean distance between nodes $i, j \in V$

Basic model: p-Center Model

Mixed-integer piece-wise linear model

$$
\begin{array}{lll}
\text{Minimize} & f(x) = \max\limits_{i,j}\left\{d_{ij}x_{ij}\right\} & \\
\text{subject to} & \sum\limits_{i\in V} x_{ij} = 1 & j \in V \\
 & \sum\limits_{i\in V} x_{ii} = p & \\
 & (1-\tau)\mu x_{ii} \leq \sum\limits_{j\in V} w_j x_{ij} \leq (1+\tau)\mu x_{ii} & i \in V \\
 & x_{ij} \in \{0,1\} & i, j \in V
\end{array}
$$

MILP model

$$
\begin{array}{lll}
\text{Minimize} & f(x) = z & \\
\text{subject to} & z \geq d_{ij}x_{ij} & i, j \in V \\
 & \sum\limits_{i\in V} x_{ij} = 1 & j \in V \\
 & \sum\limits_{i\in V} x_{ii} = p & \\
 & (1-\tau)\mu x_{ii} \leq \sum\limits_{j\in V} w_j x_{ij} \leq (1+\tau)\mu x_{ii} & i \in V \\
 & x_{ij} \in \{0,1\}, z \geq 0 & i, j \in V
\end{array}
$$

Basic model: Remarks

- Problem are NP-hard (too hard to solve)
- Real-world instances are typically very large
- Binary variables is $O(n^2)$
- Up to 80- to 100-node instances can be solved by B&B methods

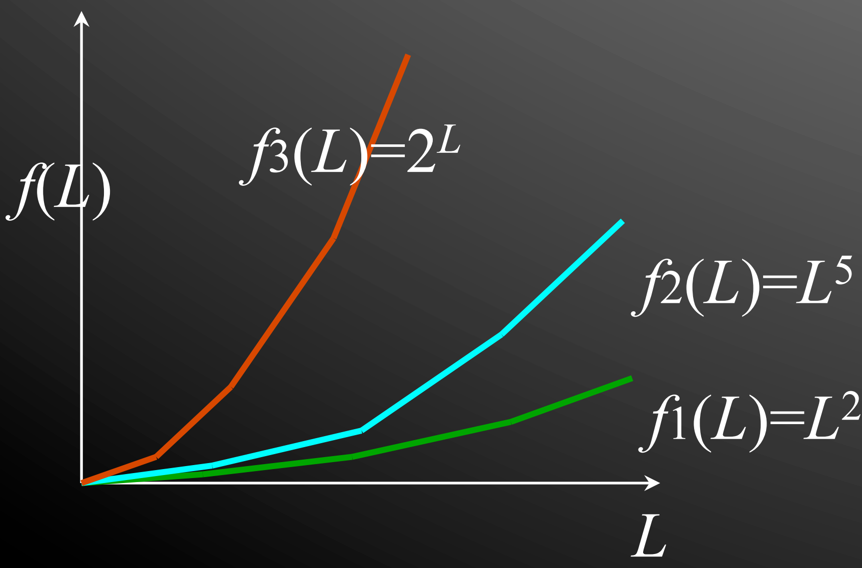
Model Issues

- How hard is the problem
- How to solve it exactly
- How to solve it approximately
- Lower bounds (model relaxations)

Model Issues

- Why is it too hard to solve??
- # of feasible partitions grows exponentially!!

Computational complexity



CPU time (1 m flops p/sec)			
L =	10	30	50
$f_1(L)=L^2$	.0001 sec	.0009 sec	.0025 sec
$f_2(L)=L^5$	.1 sec	24.3 sec	5.2 min
$f_3(L)=2^L$	.001 sec	17.9 min	35.7 years

Problem is “easy” if there is a polynomial-time algorithm

Problem is “hard” if no such algorithm exists

Model Issues: Methodolgy

- Mathematical model
- Analysis
- Derivation/development of solution technique
- Computer implementation
- Empirical evaluation

Solving Hard Optimization Problems

- Exact Methods (Enumerative Algorithms)
  - Dynamic programming
  - Branch & Bound
  - Branch & Cut & Price (advanced techniques)
- Approximate Methods (Heuristics)
  - GRASP
  - Tabu Search
  - Scatter Search
  - Simulated Annealing
- Lower bounds
  - Model relaxations
  - Lagrangian relaxation

Solving Territory Design Problems

- Exact Methods (Enumerative Algorithms)
  - Branch and bound for small problems
  - Branch-and-price for political districting
  - No polyhedral results
- Approximate Methods (Heuristics)
  - Location-allocation heuristics
  - Metaheuristics (GRASP, Tabu Search)
- Lower bounds
  - Practically nothing

Solving Problems: Exact vs. Fast

- Exact solutions may take forever to obtain
- Heuristics are in general more practical in industrial settings
- How to develop a good heuristic?
  - Exploit problem structure
  - Construction method
  - Improvement method (local search)

Basic model: p-Center Model

Minimize $f(x) = \max_{i,j} \{ d_{ij} x_{ij} \}$

subject to

$$\sum_{i \in V} x_{ij} = 1 \qquad j \in V$$

$$\sum_{i \in V} x_{ii} = p$$

$$(1-\tau)\mu x_{ii} \le \sum_{j \in V} w_j x_{ij} \le (1+\tau)\mu x_{ii} \qquad i \in V$$

$$x_{ij} \in \{0,1\} \qquad i, j \in V$$

Mixed-integer piece-wise linear model

MILP model

Minimize $f(x) = z$

subject to

$$z \ge d_{ij} x_{ij} \qquad i, j \in V$$

$$\sum_{i \in V} x_{ij} = 1 \qquad j \in V$$

$$\sum_{i \in V} x_{ii} = p$$

$$(1-\tau)\mu x_{ii} \le \sum_{j \in V} w_j x_{ij} \le (1+\tau)\mu x_{ii} \qquad i \in V$$

$$x_{ij} \in \{0,1\}, z \ge 0 \qquad i, j \in V$$

p-Center Model: Combinatorial formulation

$$\min_{X \in \Pi} f(X) = \max_{k \in K} \max_{j \in X_k} \left\{ d_{c(k),j} \right\}$$

subject to

$$(1-\tau)\mu^a \le w^a(X_k) \le (1+\tau)\mu^a \qquad k \in K, a \in A$$

$$G^k(X_k, E(X_k)) \text{ connected} \qquad k \in K$$

$$c(k) = \arg\min_{i \in X_k} \left\{ \max_{j \in X_k} d_{c(k),j} \right\}$$

where

$\Pi :=$ Collection of all p – partitions of V

$X = (X_1, \ldots, X_p) \in \Pi$

$w^a(X_k) = \sum_{j \in X_k} w_j^a$ size of X_k respect to activity a

$\mu^a = w^a(V)/p$ goal for activity a

G R A S P

G reedy

R andomized

A daptive

S earch

P rocedure

- Origin
  - Tom Feo (U. of Texas) and Mauricio Resende (U. of California), late 1980s

- Motivation
  - Constructive heuristic
  - Diverse solutions of good quality
  - TDP: Handling of contiguity

GRASP: The Basics

Basic GRASP

GRASP (N, α)
Input: N := iterations; α quality threshold
Output: Feasible solution S

$S^{best} \leftarrow \emptyset$;
for (i = 1, …, N) **do**
    S ← **GreedyRandomized**(α);
    S ← **Postprocess**(S);
    if (S better than S^{best}) **then** $S^{best} \leftarrow S$;
endfor;
return S^{best};

GRASP

- **Construction Phase:** Build q territories

    1) Greedy function

    2) Restricted candidate list (RCL)

    3) Randomized choice of best candidates

    4) Adaptive

- **Adjustment Phase:** (for p)

    Case $q > p$: Merge territories

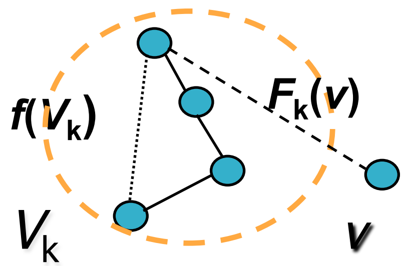
    Case $q < p$: Split territories

- **Improvement Phase:**

    1) Local search

GRASP: Construction Phase

Greedy function



Dispersion of V_k :

$$f(V_k) = \max_{i,j \in V_k} d_{ij}$$

If v is added to V_k :

Dispersion of $V_k \cup \{v\}$	Infeasibility of $V_k \cup \{v\}$	
	For $a \in A$	Total
$F_k(v) = \max\left\{\max_{j \in V_k} d_{vj}, f(V_k)\right\}$	$g_k^a(v) = \max\begin{cases} W^a(V_k \cup \{v\}) - (1+\tau^a)\mu^a \\ 0 \end{cases}$	$G_k(v) = \sum_{a \in A} g_k^a(v)$

Greedy function ➔

$$\varphi(v) = \lambda F_k(v) + (1-\lambda) G_k(v)$$

(normalized measures)

GRASP: Construction Phase

Restricted Candidate List (RCL)

Restricted by quality value α

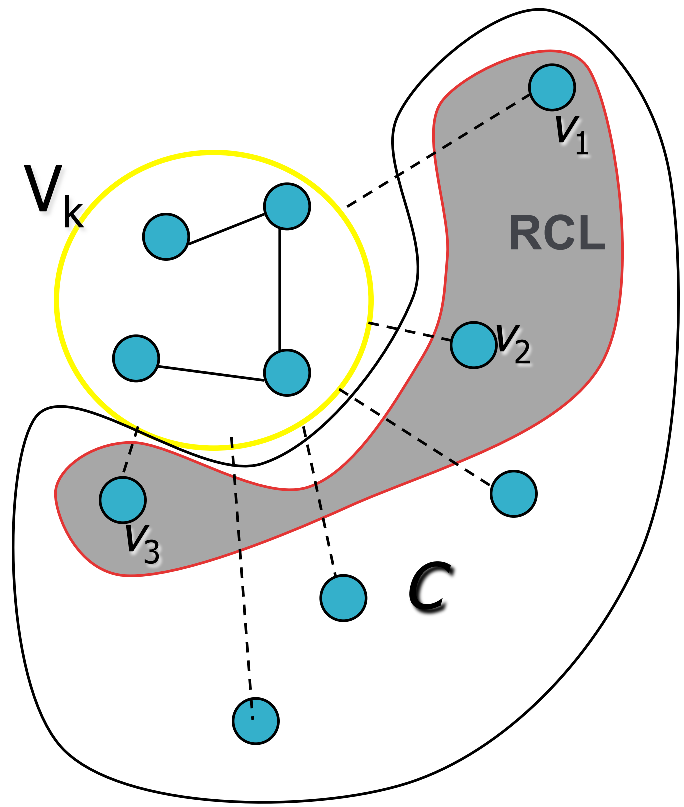
U := Set of unassigned nodes

Candidate list
C ← $\{v \in U \mid V_k \cup \{v\} \text{ is connected}\}$

$\Phi_{\mathbf{min}} = \min\{\varphi(v) : v \in C\}$

$\Phi_{\mathbf{max}} = \max\{\varphi(v) : v \in C\}$

$$\varphi(v) = \lambda F_k(v) + (1-\lambda)G_k(v)$$



RCL ←
$\{ v \in C : \varphi(v) \leq \Phi_{min} + \alpha\,(\Phi_{max} - \Phi_{min}) \}$

$\alpha = 0$ → Pure greedy
$\alpha = 1$ → Pure randomized

GRASP: Construction Phase

Adaptive Strategy

- When “to close” a territory?:
  - Size of V_k is kept below $(1+\tau^a)\mu^a \;\; \forall a$:

    $$w^a(V_k) = \sum_{v \in V_k} w_v^a \leq (1+\tau^a)\mu^a, \;\; a \in A$$
  - In practice, threshold of $\rho(1+\tau^a)\mu^a$ is used, where $\rho \in (0,1]$

- **If ($w^a(V_k) \geq \rho(1+\tau^a)\mu^a$) start a new territory**:
  - $k \leftarrow k + 1; \quad V_k \leftarrow \{ v \}$
  - v is the node of lowest degree in U
  - Build C for this new territory

- **If ($w^a(V_k) < \rho(1+\tau^a)\mu^a$) continue with current territory**:
  - Randomly choose a node v from RCL
  - $V_k \leftarrow V_k \cup \{ v \}$
  - Update neighbor list of V_k
  - Update C for current territory

GRASP

- **Construction Phase:** Build q territories

  1) Greedy function

  2) Restricted candidate list (RCL)

  3) Randomized choice of best candidates

  4) Adaptive

- **Adjustment Phase:** (for p)

  Case $q > p$: Merge territories
  Case $q < p$: Split territories

- **Improvement Phase:**

  1) Local search

GRASP: Adjustment Phase

Mission: Convert a *q*-partition into a *p*-partition

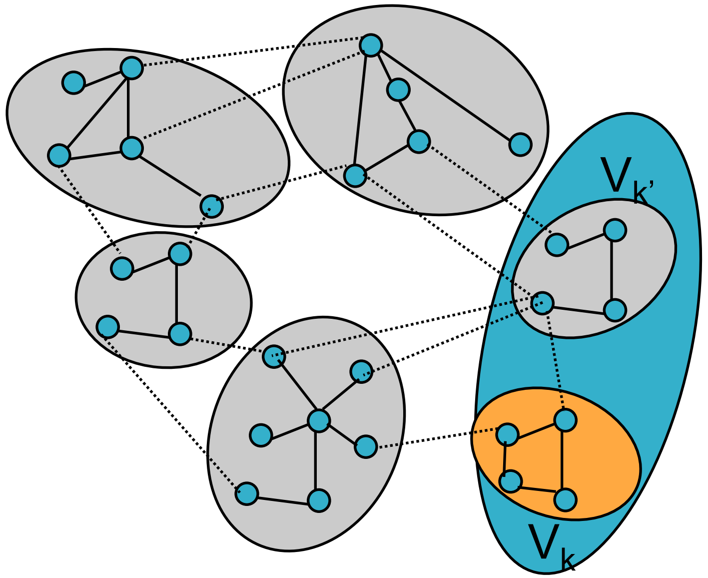
while ($q > p$)

Merge 2 territories (into 1)

- Choose V_k of lowest total weight

$$\min_{k=1,\dots,p}\left\{\sum_{a\in A} w^a(V_k)\right\}$$

- Merge it with a neighbor k' of lowest total weight



while ($q < p$)

Split "heaviest" territory into 2 territories

(In practice, very unlikely $q < p$)

GRASP

- **Construction Phase:** Build q territories

  1) Greedy function

  2) Restricted candidate list (RCL)

  3) Randomized choice of best candidates

  4) Adaptive

- **Adjustment Phase:** (for p)

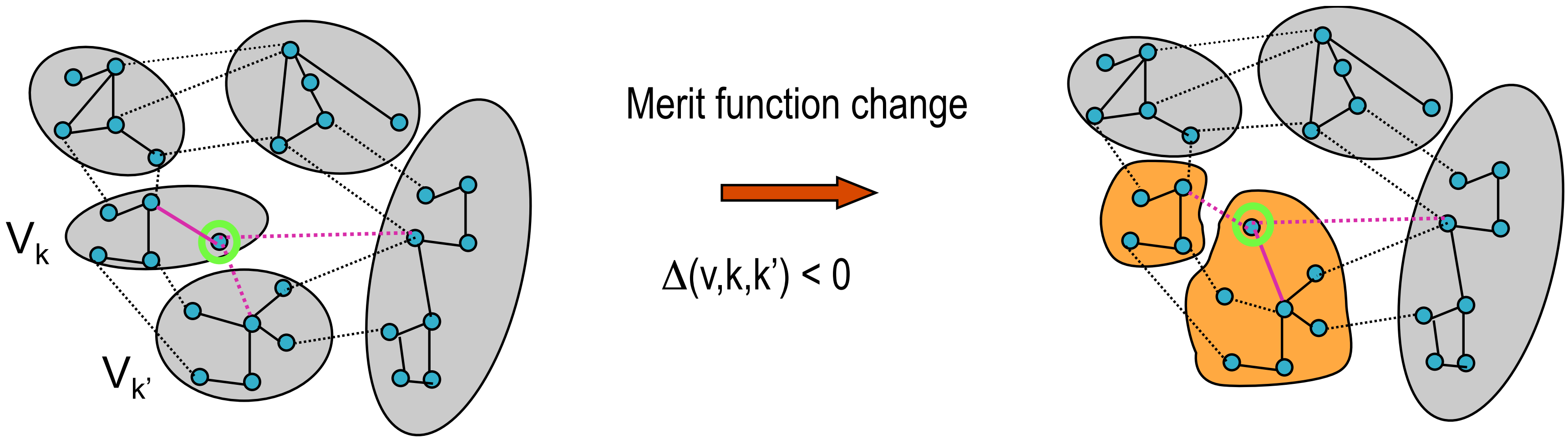
  Case $q > p$: Merge territories
  Case $q < p$: Split territories

- **Improvement Phase:**

  1) Local search

GRASP: Local Search

Solution from Construction Phase may not satisfy balance constraints



N(X): Set of solutions reachable from X = (X_1, …, X_p) by reassigning a node from X_k to a different territory

Move(i, k): Move node i (from t(i)) to territory k (with k ≠ t(i))

Weighted merit function

$$\vartheta(i,k) = \lambda \Delta f(i,k) + (1-\lambda)\Delta g(i,k)$$

where

$$\Delta f(i,k) = f(X) - f(X^{\text{new}})$$

$$\Delta g(i,k) = g(X) - g(X^{\text{new}})$$

$$X^{\text{new}} = (X_1, \ldots, X_k \cup \{i\}, \ldots, X_{t(i)} - \{i\}, \ldots, X_p)$$

$$g(X) = \sum_{k \in K, a \in A} g_k^a(X_k)$$

$$g_k^a(X_k) = \max \begin{cases} w^a(X_k) - (1+\tau^a)\mu^a \\ (1-\tau^a)\mu^a - w^a(X_k) \\ 0 \end{cases}$$

GRASP: Improving Efficiency

Basic GRASP

GRASP (N, α)
Input: N := iterations; α quality threshold
Output: Feasible solution S

$S^{best} \leftarrow \emptyset$;

for (i = 1, …, N) **do**

    S ← **GreedyRandomized**(α);

    if ($q \neq p$) **then** S ← **Adjust**(S);

    S ← **Postprocess**(S);

    if (S better than S^{best}) **then** $S^{best} \leftarrow S$;

endfor;

return S^{best};

GRASP: Improving Efficiency

GRASP with Filtering

GRASP_filter (N, α, β)

Input: N := iterations; α := quality threshold; β := filter
Output: Solución factible S

```
Sbest ← Ø;   sum ← 0;   times ← 0;
for (i = 1, …, N) do
    S ← GreedyRandomized( α );
    if (q ≠ p) then S ← Adjust( S );
    if ( i < 100  and  β (1- β ’) z(S) < z(Sbest) ) then
        S’ ← Postprocess( S );
        if (S’ better than Sbest) then Sbest ← S’ ;
        times ← times + 1;   sum ← sum + (z(S) – z(S’ )) / z(S);
        β ’ ← sum / times;
    endif;
endfor;
return Sbest;
```

β’ := average local search relative improvement

GRASP: Improving Efficiency

Reactive GRASP

- **Components:**
  - $X = \{ \alpha_1, \ldots, \alpha_m \}$:= Set of possible thresholds (α)
  - p_i := Probability associated with α_i, (=1/m at start)
  - A_i := Average value of solution found under α_i
  - z^* := Current solution
  - Periodic update of $p_i = q_i / (q_1+\ldots q_m)$ with $q_i = z^*/ A_i$
- **Advantages:**
  - High values of q_i correspond to appropriate values of α_i
  - Corresponding p_i increase when updated
  - More robust code
  - Self-tuning reduces parameter callibration

GRASP for p-Center TDP: Results

Instance Generation

- Instances randomly generated with data provided by industrial partner (20 for each combination)
- Planar maps with |V| = 1000 and 2000 in [0, 500] x [0,500]
- For each city block:
  - Number of customers → U(0, 3)
  - Demand → U(1, 12)
  - Workload → U(1, 12)
- Each BU is aggregated information of 34,000 / *n* city blocks
- $\tau^a = 0.3\,(\mathrm{DS30}), 0.2\,(\mathrm{DS20}), 0.1\,(\mathrm{DS10}), 0.05\,(\mathrm{DS05})$

GRASP for p-Center TDP: Results

Tests

- Effect of GRASP ρ (threshold for closing territories)
- Effect of local search
- Effect of GRASP β (filtering)
- Evaluation of Reactive GRASP
- Effect of GRASP λ (weight parameter)
- Comparison with current practice

Results

Evaluation of GRASP parameter ρ de GRASP: 500-node loose instances (DS30 & DS20)

500-node instances		DS30			DS20		
Statistic		ρ = 1.0	ρ = 0.8	ρ = 0.6	ρ = 1.0	ρ = 0.8	ρ = 0.6
Weighted objective function	Best	0.39	.05	0.16	0.36	0.06	0.73
	Average	1.14	.05	0.33	0.88	0.06	1.00
	Worst	1.46	.06	0.40	1.15	0.06	1.37
Dispersion measure	Best	25.91	23.88	30.11	23.63	24.46	27.80
	Average	29.41	25.05	32.81	31.33	26.44	33.07
	Worst	36.67	26.94	36.84	43.47	29.01	38.43
Sum of relative infeasibilities	Best	0.46	0.00	0.12	0.44	0.00	0.94
	Average	1.53	0.00	0.36	1.16	0.00	1.32
	Worst	2.00	0.00	0.47	1.53	0.01	1.85
CPU time (sec)	Phase 1	119.65	94.30	74.45	104.15	83.50	66.30
	Phase 2	78.60	63.60	72.30	70.00	61.55	69.10
	Total	198.25	157.90	146.75	174.15	145.05	135.40

GRASP N = 1000, GRASP α=0.3

No Local Search (Phase 3)

V_k is kept below $\boldsymbol{\rho}(1+\tau^a)\mu^a$ for all a:

$$w^a (V_k)=\sum_{v\in V_k} w_v^a \leq \boldsymbol{\rho}(1+\tau^a)\mu^a \,,\quad a\in A, \quad \boldsymbol{\rho} \text{ in } (0,1]$$

Results

Evaluation of GRASP parameter ρ de GRASP: 500-node tight instances(DS10 & DS05)

500-node instances		DS10		DS05	
Statistic		ρ = 1.0	ρ = 0.8	ρ = 1.0	ρ = 0.8
Weighted objective function	Best	0.17	0.25	0.16	1.22
	Average	0.37	0.70	0.22	1.93
	Worst	0.52	1.00	0.34	2.35
Sparsity measure	Best	25.31	25.20	28.53	28.00
	Average	30.64	32.44	33.11	34.37
	Worst	36.10	44.13	39.56	52.75
Sum of relative infeasibilities	Best	0.16	0.25	0.13	1.62
	Average	0.44	0.89	0.21	2.65
	Worst	0.65	1.31	0.37	3.26
CPU time (sec)	Phase 1	95.55	78.05	91.10	75.20
	Phase 2	64.95	65.85	62.20	67.65
	Total	160.50	143.90	153.30	142.85

GRASP N = 1000, GRASP α=0.3

No LS (Phase 3)

Results

Evaluation of filtering (GRASP parameter β) on DS10 (n = 500)

DO NOT do LS if objective_best > β(1- β')*objective_current

500-node instances		DS10		
	Statistic	GRASP	G_F(0.6)	G_F(0.8)
Sparsity measure	Best	23.60	23.72	24.04
	Average	24.53	24.58	24.97
	Worst	35.62	25.33	26.43
Sum of relative infeasibilities	Best	0.00	0.00	0.00
	Average	0.00	0.00	0.00
	Worst	0.00	0.00	0.00
Number of infeasible solutions		0	0	0
Local search #	Min	1000	270	146
	Average	1000	515	244
	Max	1000	675	289
Relative LS improvement	Average	95.37	94.20	93.34
CPU time (seg)	Phase 1	95.25	94.60	97.00
	Phase 2	64.75	64.85	64.85
	Phase 3	100.15	41.15	19.60
	Total	260.15	200.60	181.45

β' = solution quality average improvement of the local search

GRASP N= 1000,
GRASP α=0.3,

GRASP ρ =1.0

LS_move_limit=200,
LS strategy=FF

Results

Evaluation of filtering (GRASP parameter β) on DS05 (*n* = 500)

DO NOT do LS if objective_best > β(1- β')*objective_current

500-node instances		DS05			
	Statistic	GRASP	G_F(0.4)	G_F(0.6)	G_F(0.8)
Sparsity measure	Best	24.82	23.56	24.09	23.83
	Average	26.59	26.66	27.20	26.85
	Worst	31.20	31.40	34.99	30.91
Sum of relative infeasibilities	Best	0.00	0.00	0.00	0.00
	Average	0.00	0.00	0.00	0.01
	Worst	0.01	0.02	0.01	0.04
Number of infeasible solutions		2	2	2	8
Local search #	Min	1000	392	219	153
	Average	1000	641	272	197
	Max	1000	904	329	224
Relative LS improvement	Average	93.02	91.17	88.03	85.00
CPU time (seg)	Phase 1	90.80	90.80	90.80	90.80
	Phase 2	62.85	62.85	62.85	62.85
	Phase 3	113.75	64.10	24.20	17.90
	Total	267.40	217.75	177.85	171.55

β' = solution quality average improvement of the local search

GRASP N= 1000,
GRASP α=0.3,

GRASP *ρ* =1.0

LS_move_limit=200,
LS strategy=FF

Results

Evaluation of GRASP quality parameter α

DS10 & DS05 (*n* = 500)

500-node instances Statistic	DS10 α=0.1	 α=0.2	 α=0.3	 α=0.4	 α=0.5
Average relative deviation from best	2.19	1.49	2.58	2.72	3.42
Worst relative deviation from best	6.92	5.31	5.81	5.79	10.44
Number of infeasible solutions	0	0	0	0	0
Number of best solutions	7	6	5	3	2
500-node instances Statistic	**DS05** α=0.1	 α=0.2	 α=0.3	 α=0.4	 α=0.5
Average relative gap (*)	3.52	3.89	3.46	3.03	4.89
Worst relative gap (*)	12.06	9.18	9.15	13.40	9.87
Number of infeasible solutions	2	2	1	0	0
Number of best solutions	5	3	4	7	2
(*)Excludes instances d500-06 and d500-15					

GRASP N = 1000, GRASP ρ =1.0, LS_move_limit=200, LS strategy=FF

Results

Evaluation of Reactive GRASP on DS10 & DS05 (*n* = 500)

DS10 *n* = 500	GRASP					R-GRASP
Statistic	α=0.1	α=0.2	α=0.3	α=0.4	α=0.5	RG
Average	2.58	1.87	2.97	3.10	3.81	2.29
Worst	6.92	5.31	6.69	6.17	10.44	7.78
No. best	7	5	4	1	2	5
Failed	0	0	0	0	0	0

DS05 *n* = 500	GRASP					R-GRASP
Statistic	α=0.1	α=0.2	α=0.3	α=0.4	α=0.5	RG
Average	3.77	4.13	3.70	3.28	5.14	3.81
Worst	12.06	9.18	9.15	13.40	9.87	10.69
No. best	4	3	3	7	3	4
Failed	2	2	1	0	0	1

GRASP N = 1000,

GRASP ρ =1.0,

LS_move_limit=200,

LS strategy=FF

Results

Evaluation of λ on 1000- and 2000-node instances

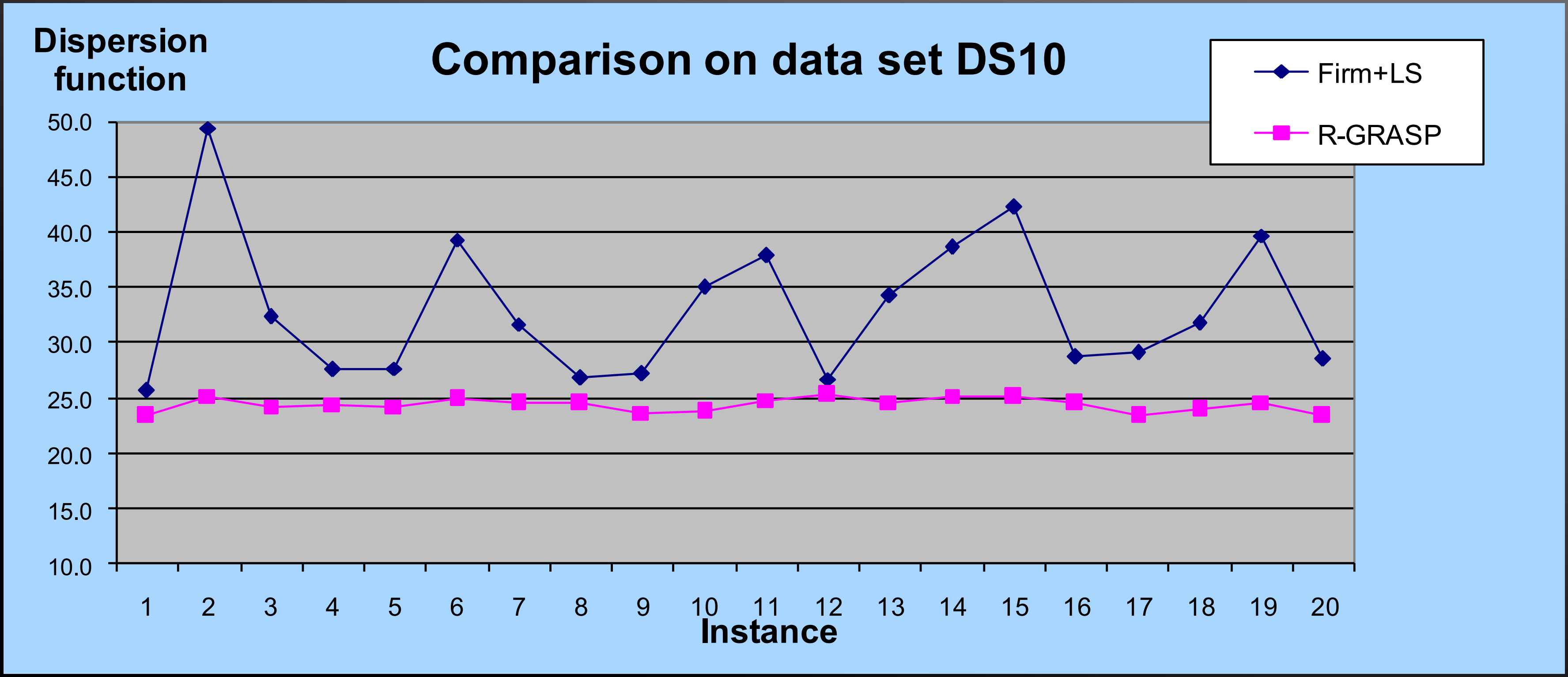
DU05 n=1000	R-GRASP				
Statistic	$\lambda=0.3$	$\lambda=0.5$	$\lambda=0.7$	$\lambda=0.9$	$\lambda=1.0$
Average gap	5.83	4.78	3.69	0.08	32.15
Worst gap	13.91	11.69	12.77	0.89	67.11
No. best	0	1	2	17	0
Failed	**0**	**0**	**0**	**0**	**20**
LS benefit (%)	**97.1**	**95.1**	**90.8**	**75.4**	**7.7**

DU05 n=2000	R-GRASP				
Statistic	$\lambda=0.3$	$\lambda=0.5$	$\lambda=0.7$	$\lambda=0.9$	$\lambda=1.0$
Average gap	7.21	4.82	3.85	2.94	29.80
Worst gap	16.61	10.45	18.83	12.05	69.58
No. best	3	3	7	7	0
Failed	**0**	**0**	**0**	**0**	**20**
LS benefit (%)	**73.9**	**72.4**	**69.0**	**55.0**	**6.0**

GRASP N = 500,
GRASP ρ =1.0,
LS_move_limit=200,
LS strategy=FF

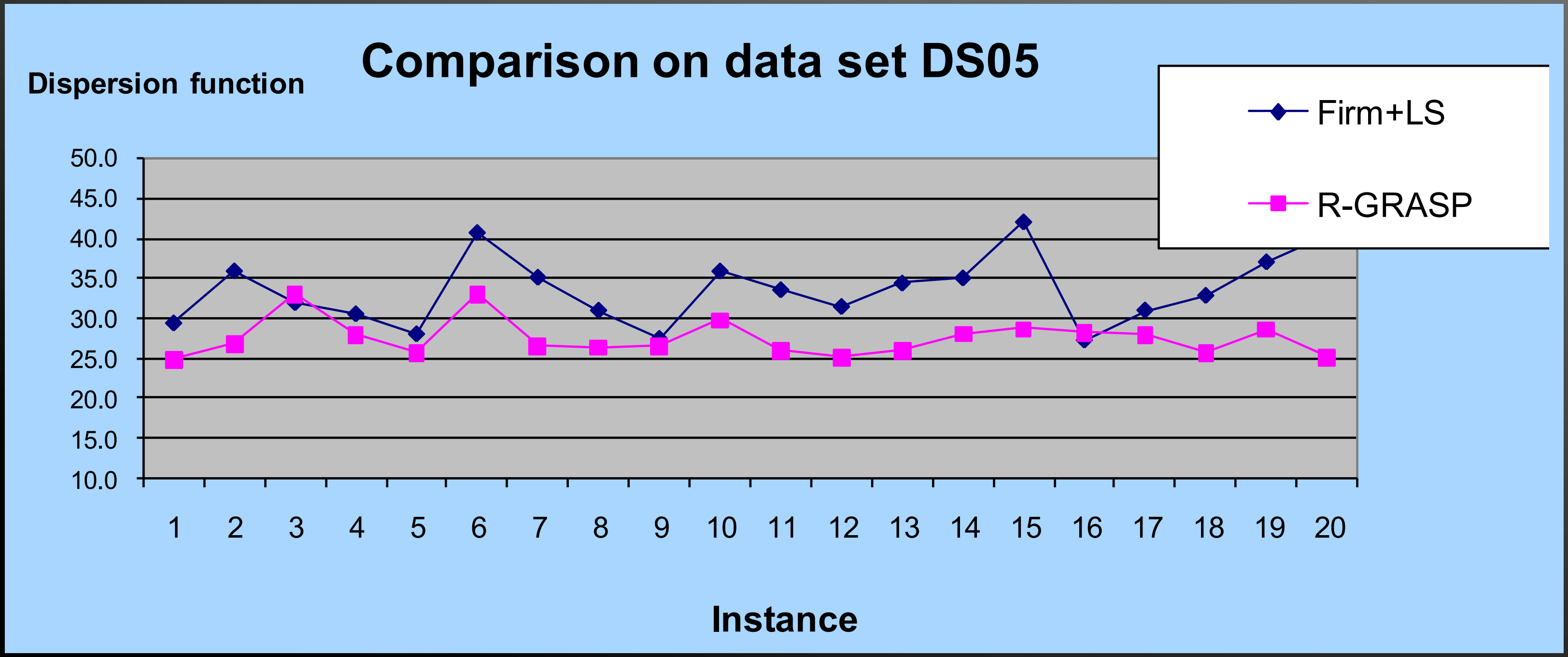
Results

Comparison with current practice (*n* = 500)



Results

Comparison with current practice (n = 500)



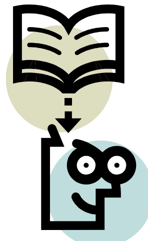
Wrap-up

- Highlights:
  - Many applications of OR (territorial systems, manufacturing, services, transportation, logistics, telecommunications, medical, natural resources, etc.)
  - All the require string COMPUTATIONAL background for software developing
  - Don’t underestimate the power of Math (both math and computing can be an amazing killing machine)
  - Learning higher-quality skills → Go to graduate school !!!

References

- B. Bozkaya, E. Erkut, and G. Laporte (2003). A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research*,144(3):12–26.
- S. W. Hess and S. A. Samuels (1971). Experiences with a sales districting model: Criteria and implementation. *Management Science*, 18:41–54.
- E. Fernández, J. Kalcsics, S. Nickel, and R. Z. Ríos-Mercado. A novel maximum dispersion territory design model arising in the implementation of the WEEE-directive. *Journal of the Operational Research Society,* 61(3):503–514.
- J. Kalcsics, S. Nickel, and M. Schröder (2005). Towards a unified territorial design approach – Applications, algorithms and GIS integration. *TOP,* 13(1):1–56.
- R. Z. Ríos-Mercado and E. A. Fernández (2009). A reactive GRASP for a commercial territory design problem with multiple balancing requirements. *Computers & Operations Research*,36(3):755–776.

Thank you …



http://yalma.fime.uanl.mx/~roger

roger@yalma.fime.uanl.mx

Acknowledgements:

Mexican Council for Science and Technology (CONACYT grant 48499-Y)

U A N L (PAICYT grant CA1478-07)

Acknowledgements:

My research assistants

Biali

Shaní

Vande