# A good recipe for solving MINLPs

Leo Liberti[1], Giacomo Nannicini[1], Nenad Mladenović[2]

[1] *LIX, École Polytechnique, France*
[2] *Brunel University, London, UK*, and
*Institute of Mathematics, Academy of Sciences, Belgrade, Serbia*

31 October 2008

# Summary of Talk

1. The Problem
2. The Ingredients
   - VNS
   - Local Branching
   - Branch-and-Bound for cMINLPs
   - Sequential Quadratic Programming
3. The RECIPE
4. MINLPLib
5. Computational Experiments
6. Conclusions

# Problem Classes

- We distinguish four classes of mathematical programs:
  1. Linear Programs (LPs)
  2. Mixed-Integer Linear Programs (MILPs)
  3. Nonlinear Programs (NLPs)
  4. Mixed-Integer Nonlinear Programs (MINLPs)
- For nonlinear programs, convexity of objective function and constraints is also important to determine the problem's difficulty

# A Difficult Task

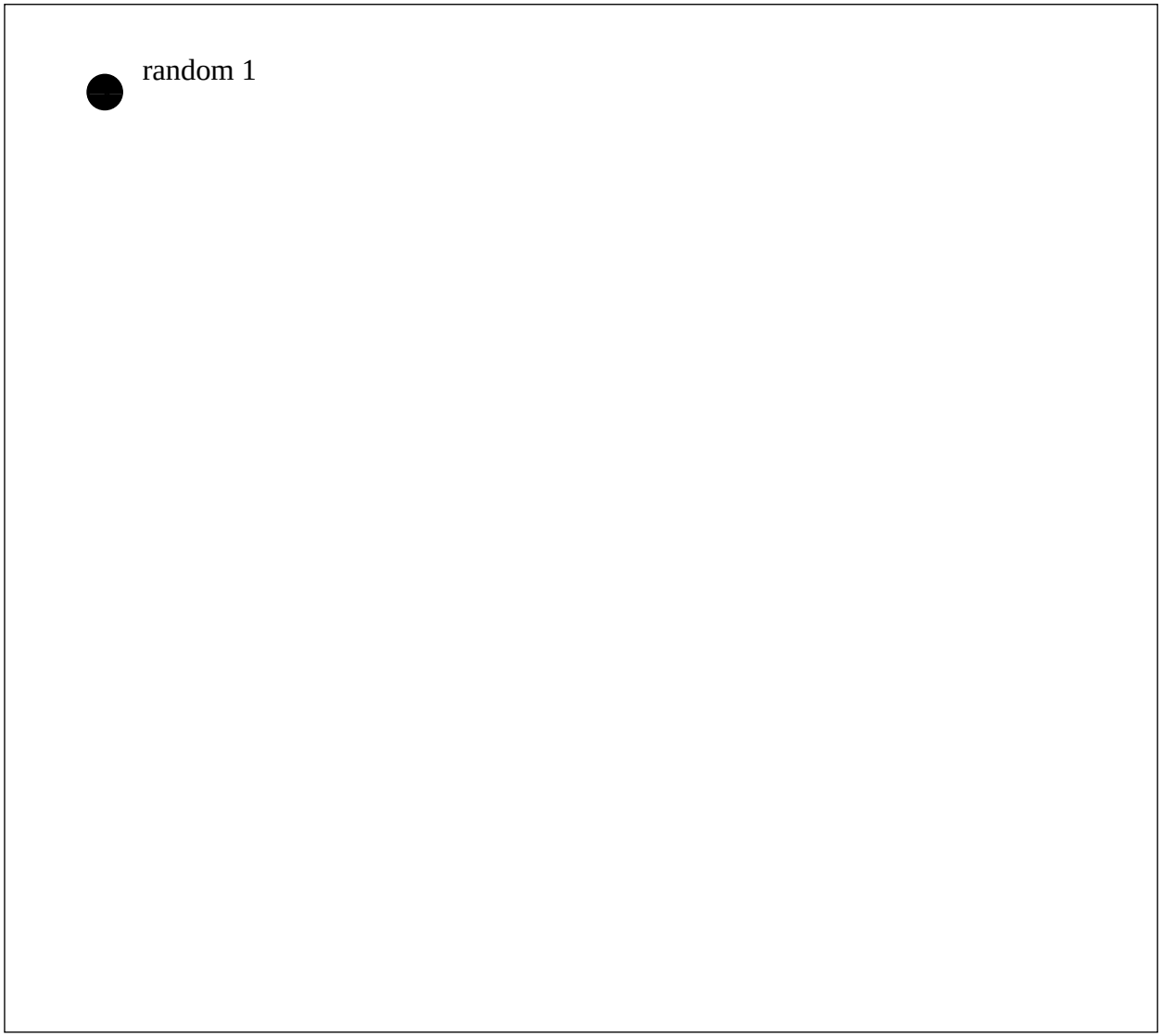
- We address the most difficult problem class: nonconvex MINLPs
- Obviously, this class is also the most expressive
- Difficulties arise from both nonconvexity and integrality
- We cannot aim for optimality, hence we would easily settle for a *fast* and *reliable* heuristic (who would not?)
- Especially true for industrial applications: we need something that is easy to use and does not need hundreds of parameters

# VNS: Introduction

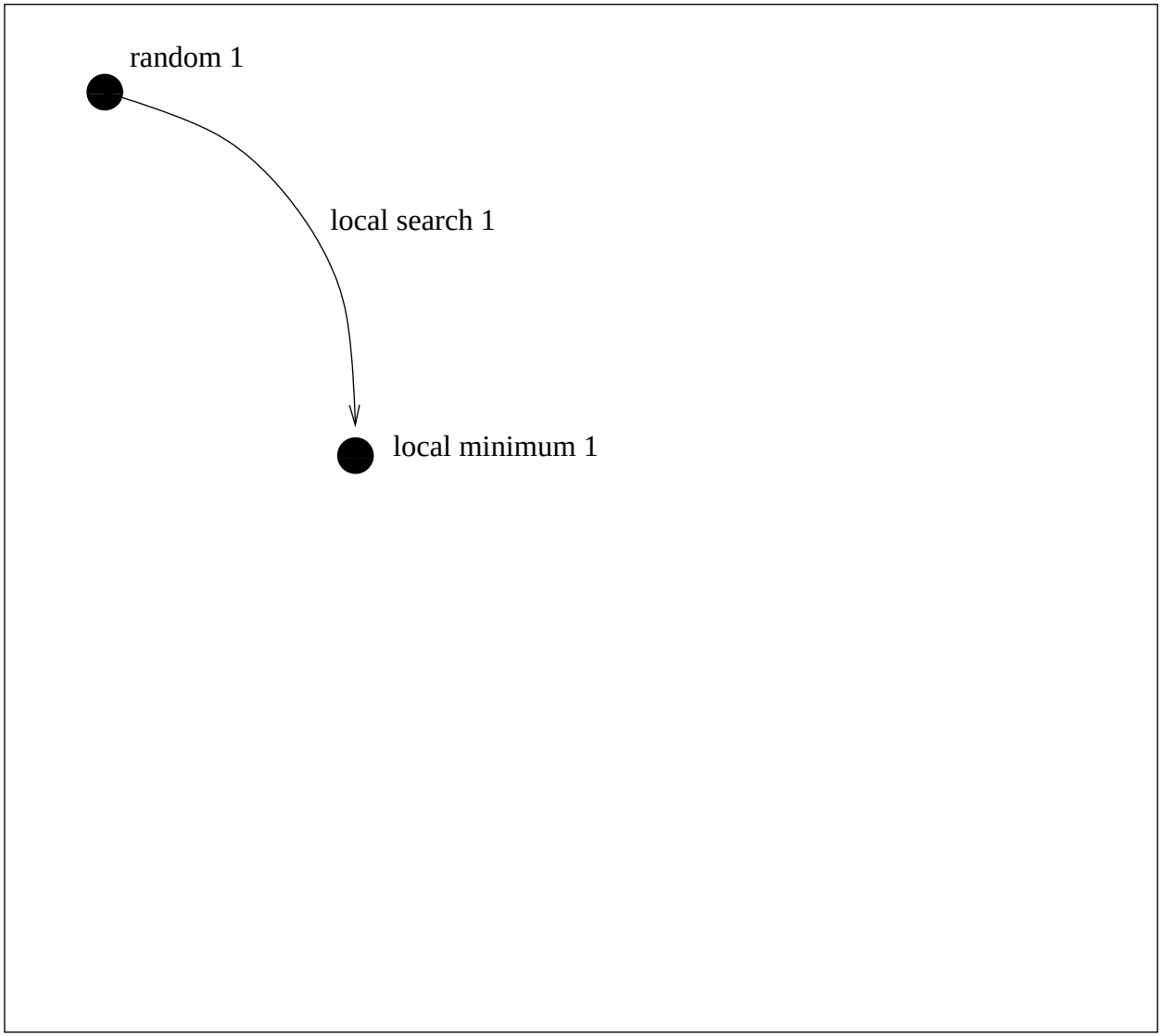
- Applicable to discrete and continuous problems
- Uses any local search as a black-box
- In its basic form, easy to implement
- Few configurable parameters
- Used for a variety of problem, large amount of references [Hansen and Mladenović, 2001]

# VNS: Algorithm (sketch)

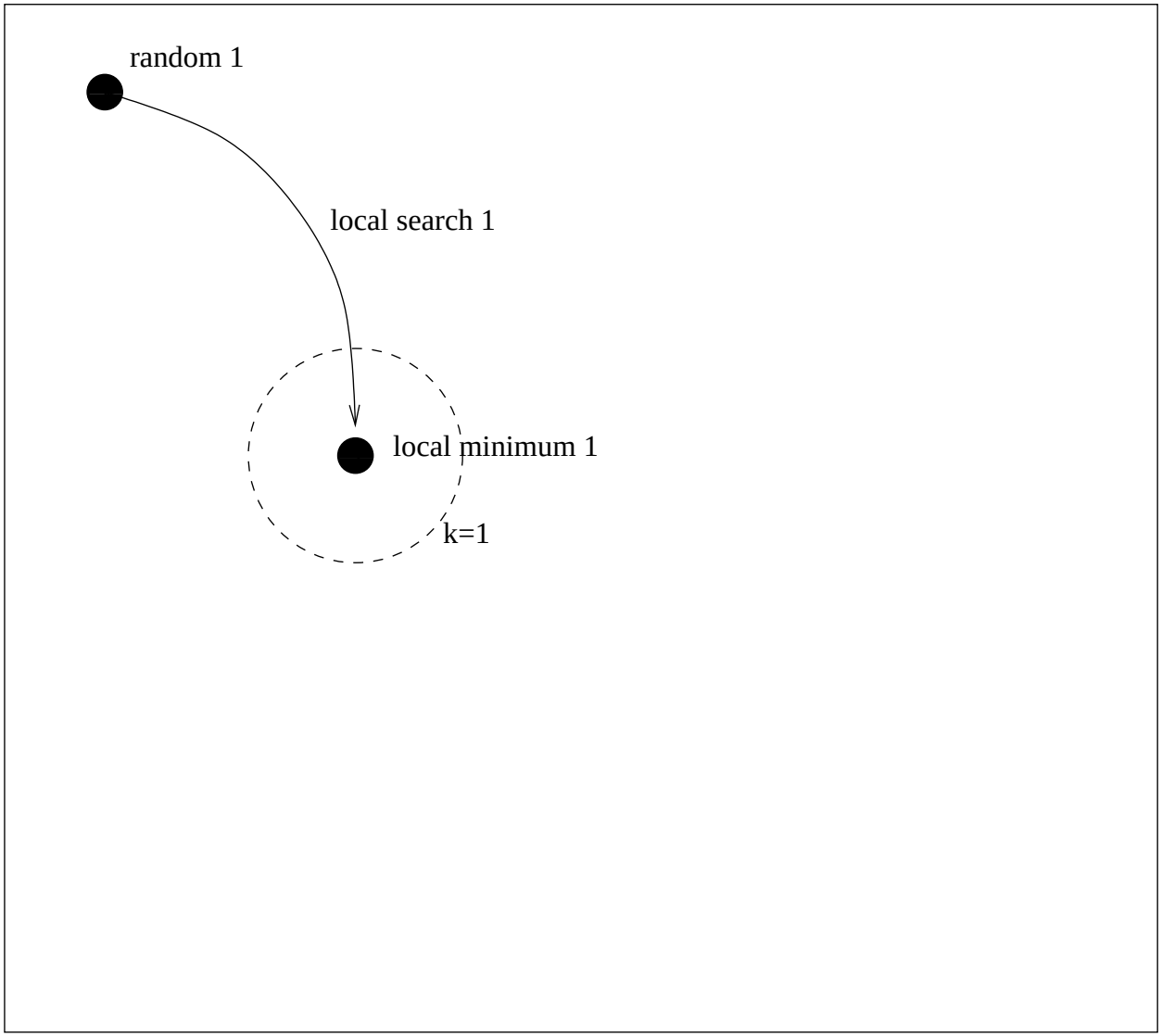
# VNS: Algorithm (sketch)



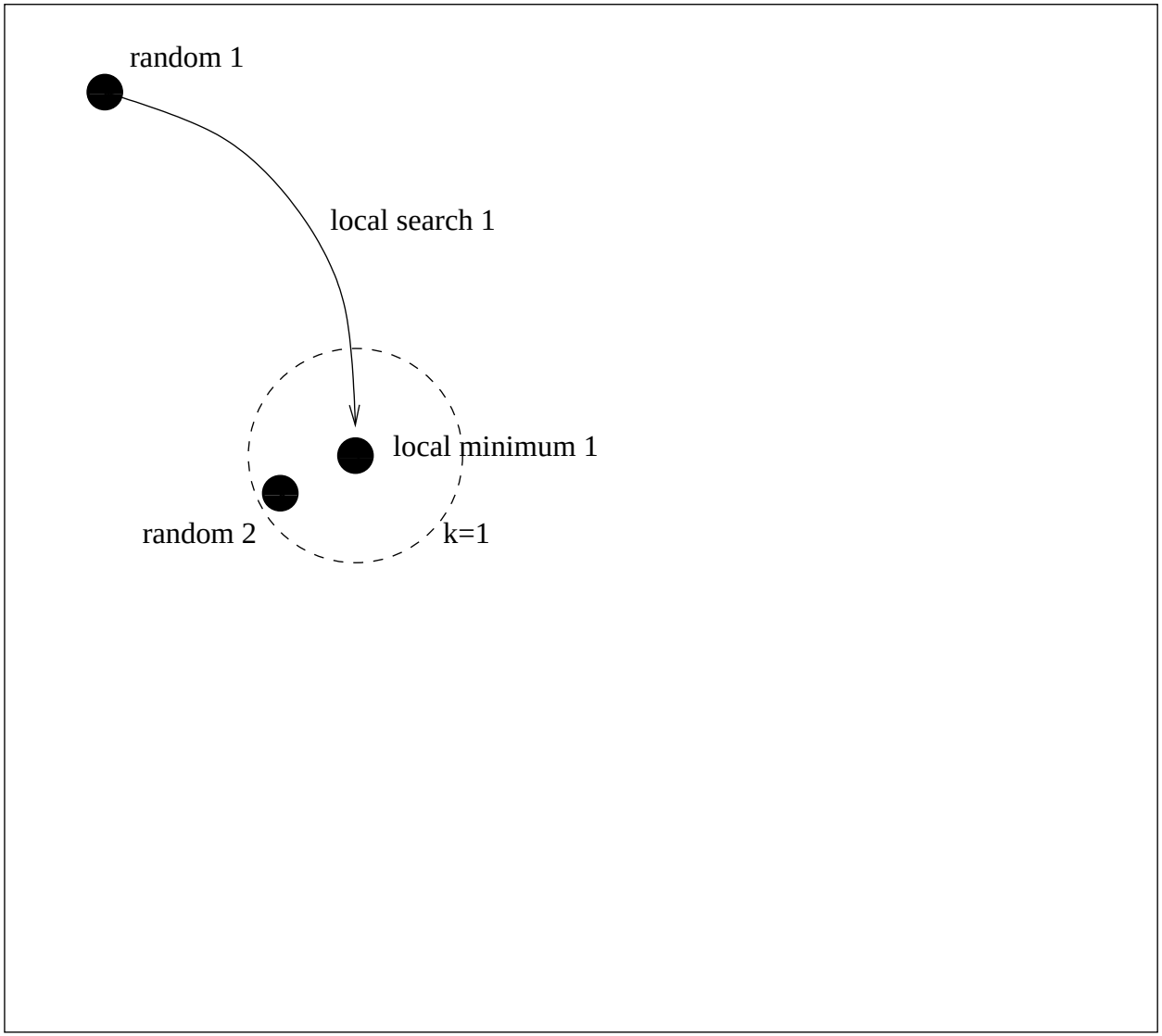
# VNS: Algorithm (sketch)



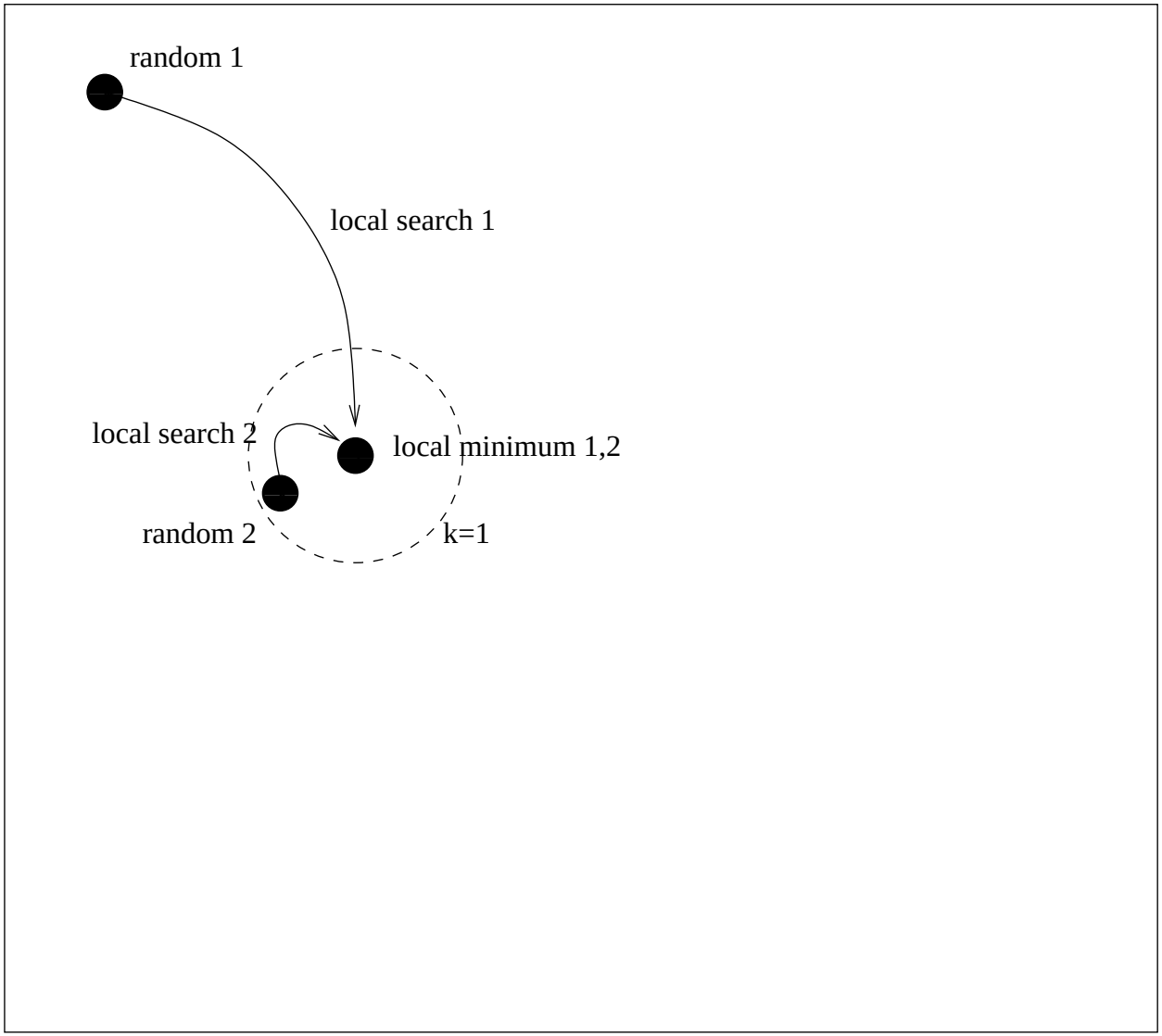
# VNS: Algorithm (sketch)



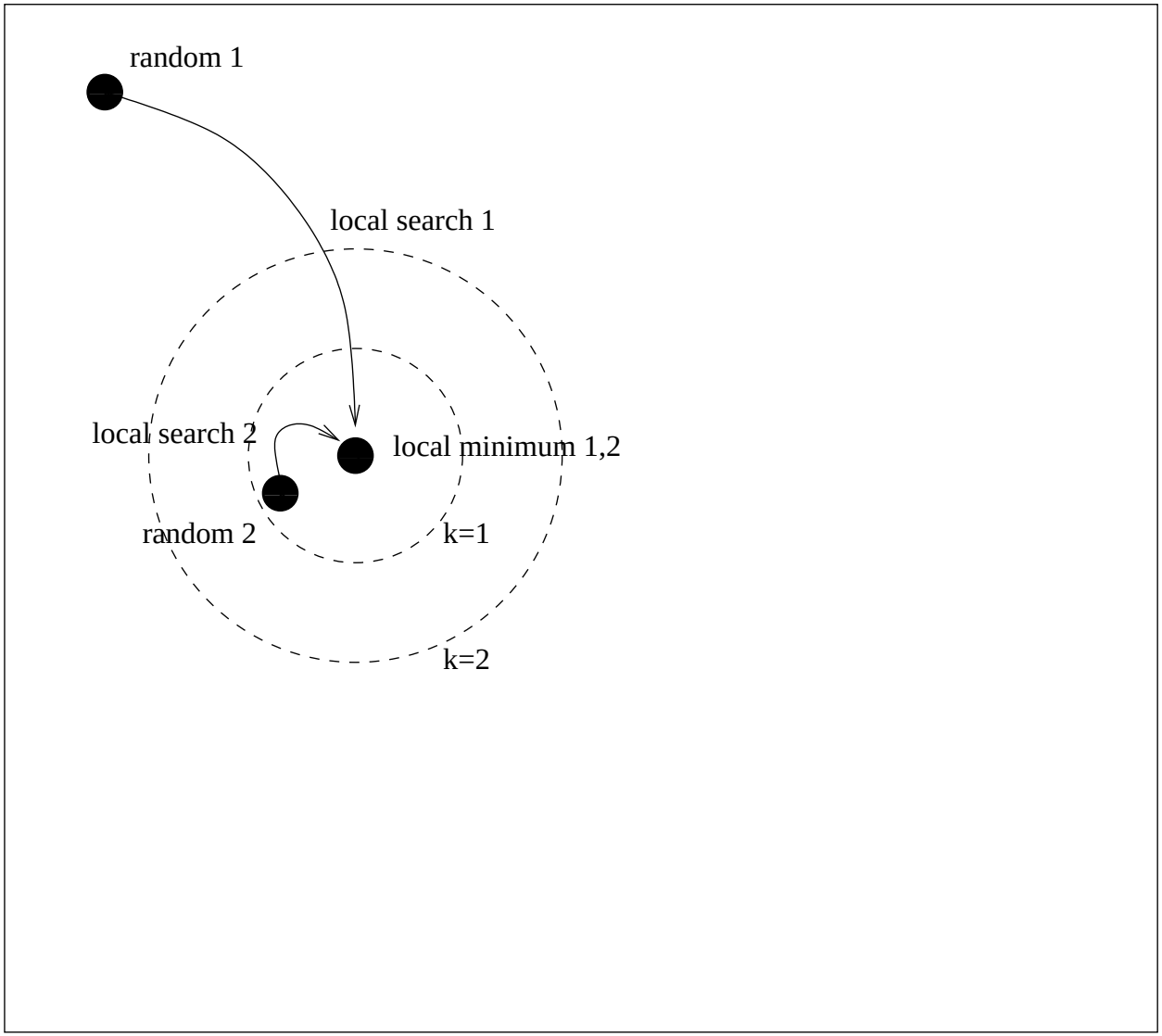
# VNS: Algorithm (sketch)



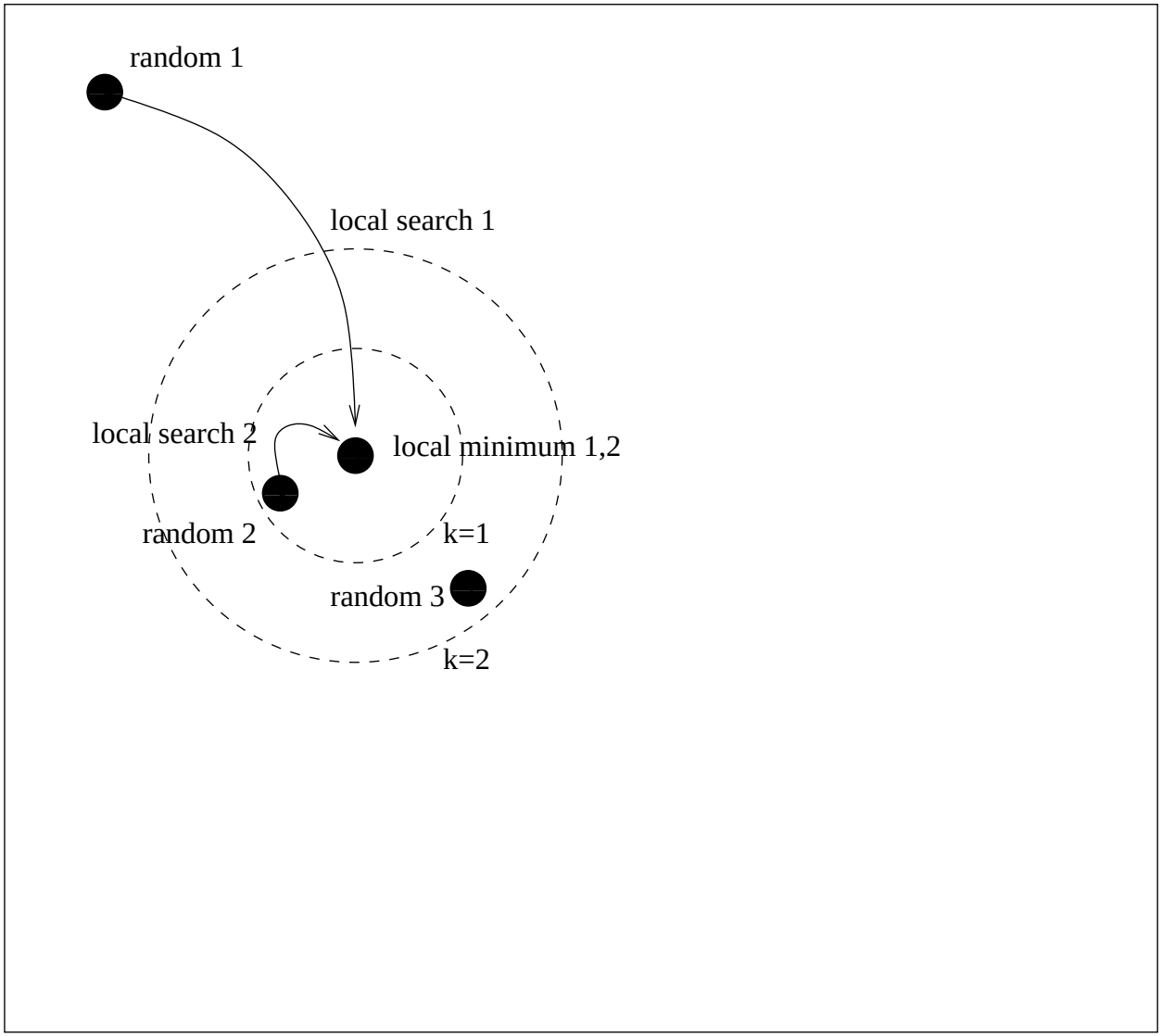
# VNS: Algorithm (sketch)



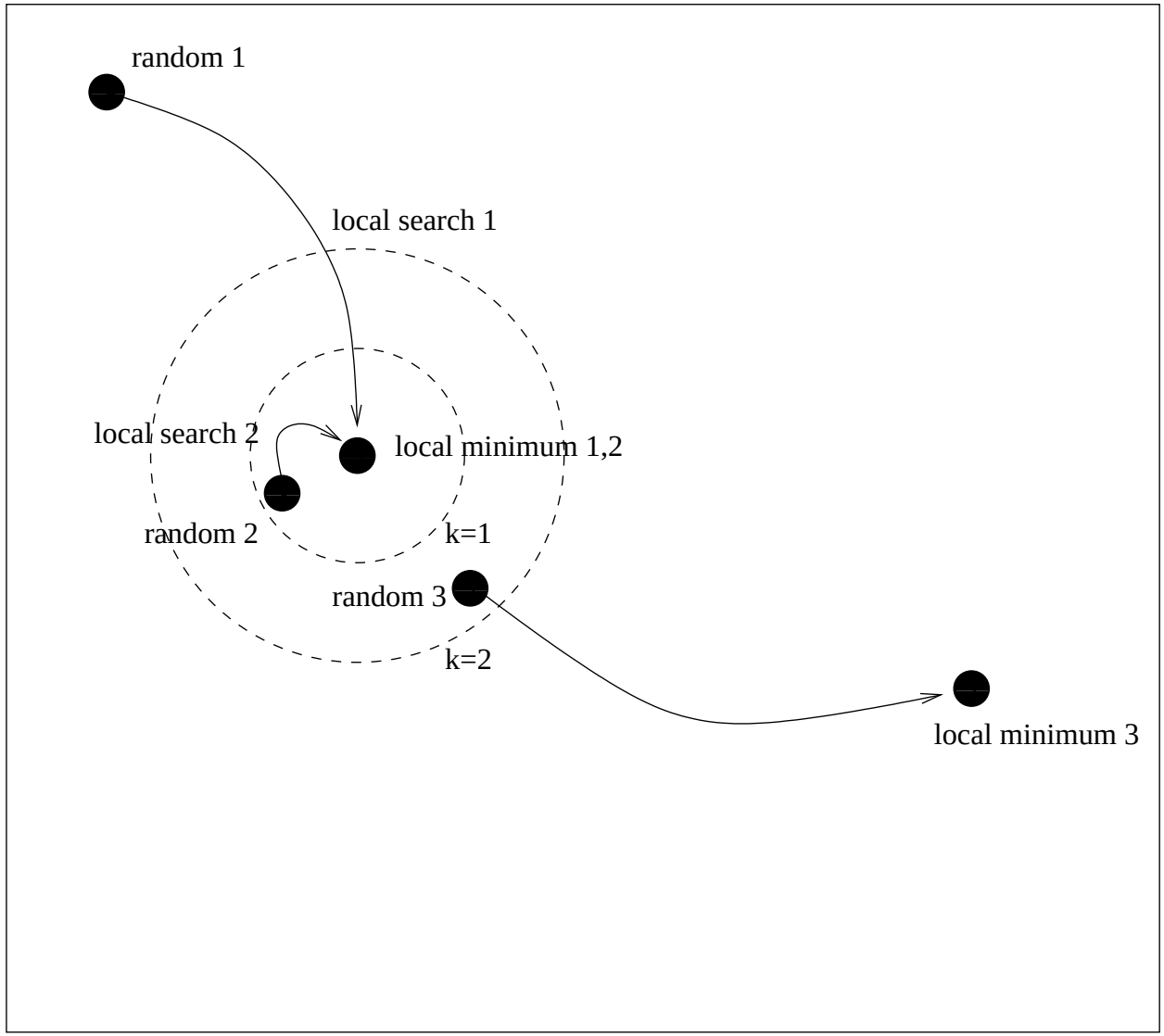
# VNS: Algorithm (sketch)



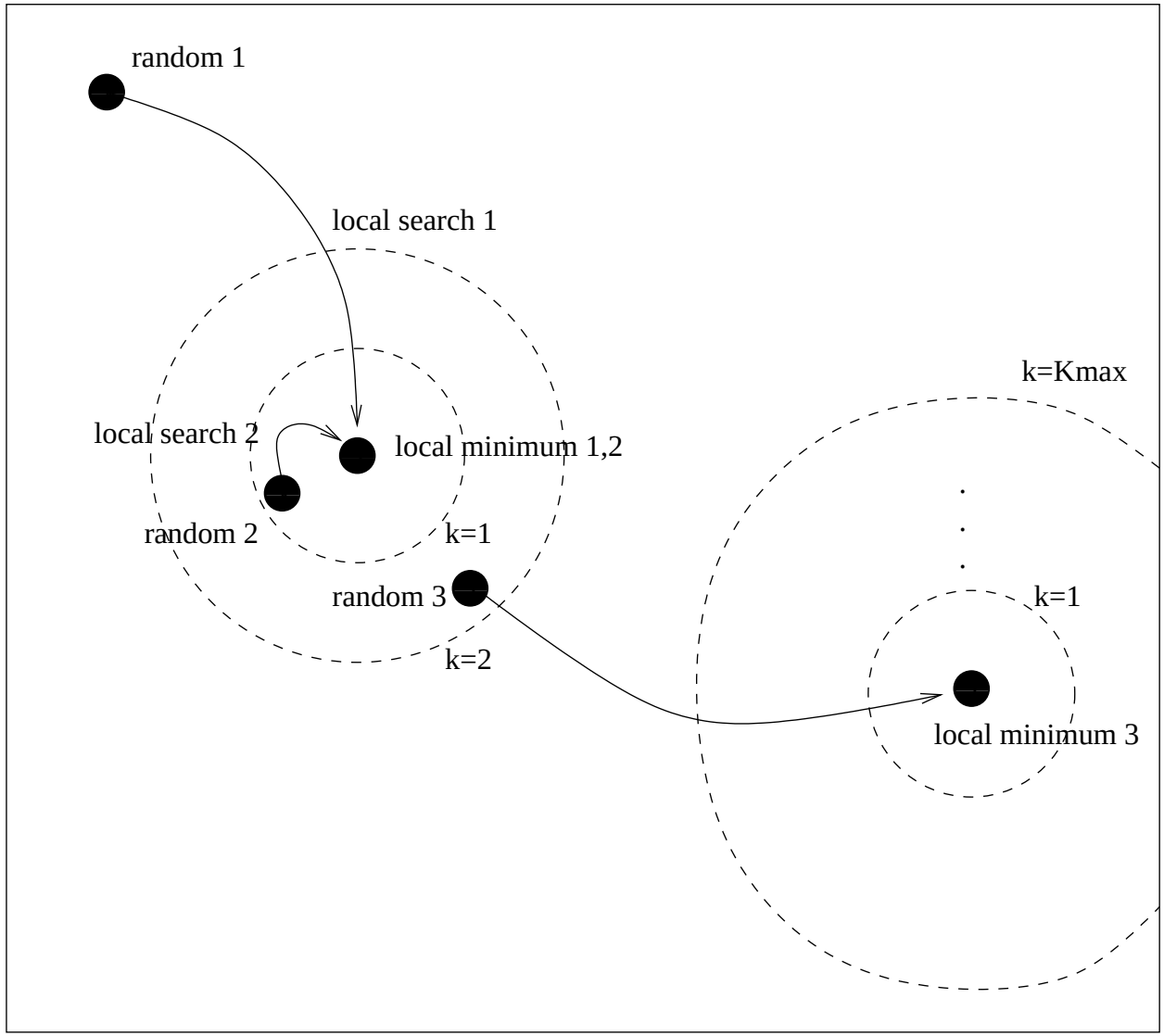
# VNS: Algorithm (sketch)



# VNS: Algorithm (sketch)



# VNS: Algorithm (sketch)



# VNS: Configurable parameters

- Maximum neighbourhood size: $k_{\max}$
- Number of local searches in each neighbourhood $L$
- Other parameters if further termination conditions included, e.g.:
  - maximum running time
  - desired objective function value
  - ...

# Local Branching

- Efficient heuristic to find good solution for MILPs.
- We consider Local Branching for binary variables [Fischetti and Lodi, 2005]
- Let $B$ be the set of indices of binary variables, let $x^*$ be a binary feasible solution, and let $k \in \mathbb{Z}$, $k > 0$
- We explore the $k$-neighbourhood of $x^*$ by enforcing the local branching constraint:

$$\sum_{i \in B : x_i^* = 1} (1 - x_i) + \sum_{i \in B : x_i^* = 0} x_i \leq k$$

# Branch-and-Bound for cMINLPs

- For convex problems, we can use Branch-and-Bound methods, since obtaining lower bounds is easy
- A typical approach is to relax integrality and solve the resulting convex NLP to optimality
- Branching is done on integer variables
- If the problem is nonconvex, BB can be used as an heuristic (i.e., it finds a local optimum)
- The BB local NLP subsolver needs an initial feasible starting point
- We supply such point computing a constraint feasible (possibly non integral!) solution with an SQP method; this increases our chances of finding an *integer* feasible point

# Sequential Quadratic Programming

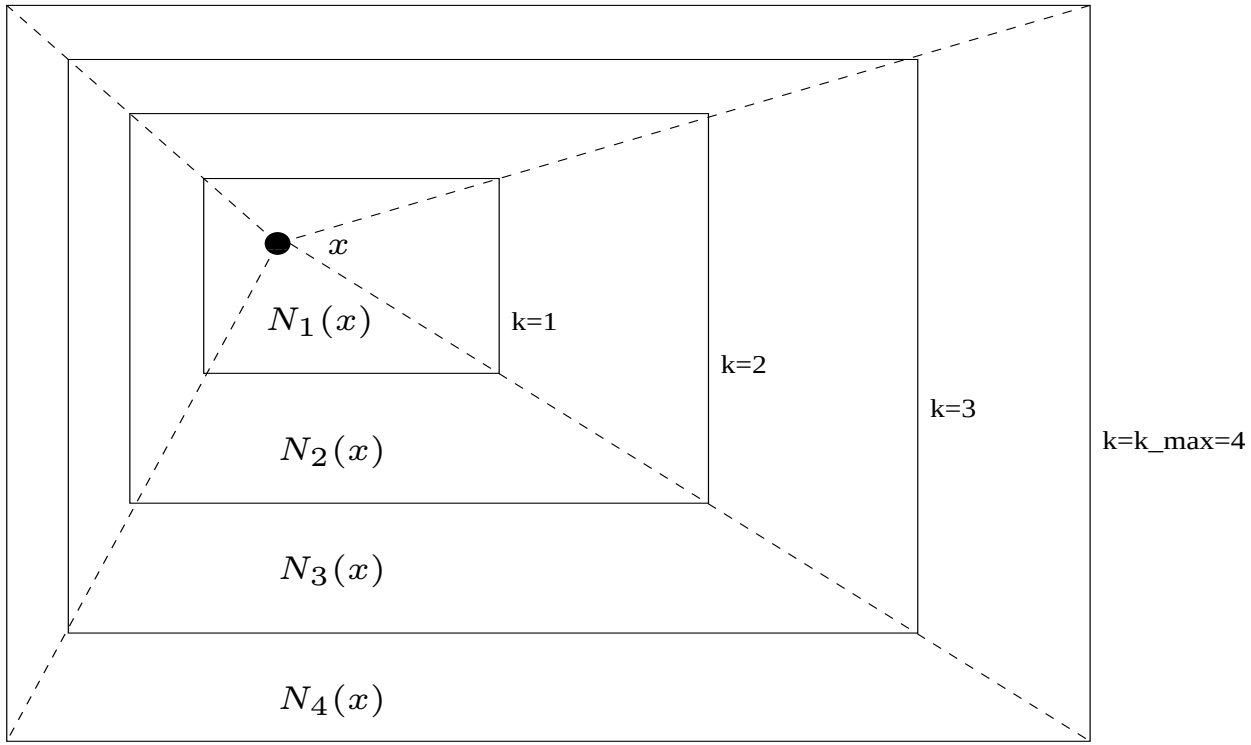
- SQP methods find local optima to nonconvex MINLPs
- Idea: solve a sequence of quadratic approximation of the original problem subject to a linearization of the constraints
- The quadratic approximation is obtained by a convex model of the objective function Hessian at a current solution point
- SQP are now at a very advanced stage: there are good chances of finding a constraint feasible point starting from any given initial point, even for large NLPs

# The RECIPE algorithm

- We put together all the ingredients into our Relaxed-Exact Continuous-Integer Problem Exploration using a two-phase search
  1. Global search: VNS
  2. Local search: BB method, with initial starting point provided by an SQP method
- Neighbourhood structure:
  - Hyper-rectangular neighbourhoods for continuous and general integer variables
  - Local branching for binary variables (note: the LB constraint is dropped for the SQP search)

# Hyper-rectangular Neighbourhood

- Use hyper-rectangular neighbourhoods $N_k(x')$ proportional to the region delimited by the variable ranges
- In other words, hyper-rectangular shells of size $k/k_{\max}$ of the original domain



# Local Branching Neighbourhood

- Hyper-rectangular neighbourhoods are ineffective for binary variables
- Each variable would keep the same value for $k \leq k_{\max}/2$, and all variables would be unconstrained for $k > k_{\max}/2$
- Hence, we use the local branching constraint with a rhs depending on the size of the current neighbourhood $k$:

$$\sum_{i \in B : x_i^* = 1} (1 - x_i) + \sum_{i \in B : x_i^* = 0} x_i \leq |B| \frac{k}{k_{\max}}$$

# MINLPLib

- Collection of MINLPs, from various sources:
  - Small-scale problems from the literature
  - Large-scale industrial problems
- Provides a common testset for MINLP solvers
- Available at
  `http://www.gamsworld.org/minlp/minlplib.htm`
- The website provides the best known optimum for each instance (and lists the solvers that obtained that value)

# MINLPLib

- Total number of instances: 265
  - Smallest instance: 1 constraint, 3 variables (2 integer)
  - Largest instance: 24972 constraints, 23827 variables (10920 binary)
- Instances are in GAMS format, but an automatic AMPL reformulator is available
- 20 instances had to be removed due to problems in the reformulation (i.e., unknown or unimplemented operators) $\longrightarrow$ 245 instances

# Implementation

- VNS: AMPL scripting language [Fourer and Gay, 2002]
- BB solver: `minlp_bb` [Leyffer, 1999]
- SQP solver: `snopt` [Gill, 2006]
- We also used a C program to reformulate the problems with the local branching constraints

# Parameters

- We tested RECIPE on 245 instances of the MINLPLib
- We used the same parameters for all runs:
  - $k_{\max} = 50$
  - $L = 15$
  - Max CPU time: $10$ hours
- As we always use the same random seed, we did only one run on each instance

# Results: Feasibility

- Out of 245 instances:
  - 163 putative global optima found ($66.3\%$)
  - 82 with no feasible point found ($33.7\%$)
- Failures were due to both local solvers
  - Several difficult instances where `snopt` does not find any feasible point, or simply fails
  - On small instances, most failures are due to `minlp_bb`, which is unable to attain integrality

# Failures: Statistics

- We compared the average number of variables, constraints, nonzeroes, and nonlinear nonzeroes, as well as ratio of continuous, binary and integer variables (geometric averages)

| | # Variables | | | | # | Nonzeroes | |
|---|---|---|---|---|---|---|---|
| | Tot | Cont | Bin | Int | Cstr | Tot | NL |
| All | 73.69 | 0.487 | 0.209 | 0.212 | 73.22 | 326.45 | 50.05 |
| Solved | 46.58 | 0.447 | 0.237 | 0.219 | 41.21 | 175.41 | 34.36 |
| Unsolved | 218.57 | 0.588 | 0.144 | 0.196 | 285.18 | 1441.75 | 121.84 |

# Failures: Statistics

- We ran `snopt` on the unsolved instances (i.e. no feasible solution found)
- Out of the 82 instances:
  - Solved[1]: 56
  - Failures: 8
  - Infeasible: 18
- For the instances where no continuous feasible solution is found, the BB solver `minlp_bb` obviously fails

[1] We include instances where a feasible but not necessarily optimal solution was found

# Results: Optimality

- Out of the 163 instances where RECIPE found a feasible solution:
  - 121 known putative global optima ($74.2\%$)
  - 3 unbounded instances[2] ($1.8\%$)
  - 13 new putative global optima ($8.0\%$)
  - 26 local optima, i.e. a better solution is known ($16.0\%$)
- New optima for the following instances:
  - `csched2a`: 138 constraints, 233 variables
  - `ex3, ex3pb`: 32 constraints, 33 variables
  - `lop97icx`: 88 constraints, 987 variables
  - `nuclear14b, nuclear24b`: 1786 constraints, 1569 variables
  - `nuclear25`: 1304 constraints, 1679 variables
  - `nuclearvc, nuclearvd, nuclearvf`: 318 constraints, 352 variables
  - `nvs02, nvs14`: 4 constraints, 9 variables

[2] `minlphix, risk2b, risk2bpb`

# Results: Optimality

- In total, for $84\%$ of the instances with feasible solution we found an optimum better or equal to the best known
- This corresponds to $55\%$ of the 246 initial instances
- Ranking of methods by number of putative global optima found[3]:
  1. RECIPE: $55\%$
  2. SBB+CONOPT: $37\%$ [Drud, 1985]
  3. BARON: $15\%$ [Sahinidis and Tawarmalani, 2005]
  4. AlphaECP: $14\%$ [Westerlund and Pörn, 2002]
- Very reliable method, without changing *any* parameter!

[3] From the GAMS website

# Conclusions

- Combination of several off-the-shelf components into a reliable MINLP solver
- Global search: straightforward VNS
- Local search: SQP and BB method for convex MINLPs
- Neighbourhood structure: hyper-rectangles and local branching
- Extensive computational testing with hard MINLPs, showing good results

# Future Research: Different Solvers

- Preliminary results show that RECIPE using `ipopt` and `bonmin` is slower, but *significantly* more reliable in terms of feasibility
- Moreover, we do *not* drop the local branching constraint for the SQP search
- The algorithm takes more time, but seems to find better solutions, in particular on the instances with no binary variables

# Future Research: Bound Tightening

- We employ `coutight`, an open-source utility to tighten the variable bounds using expression trees (and more)
- Whenever we define a new neighbourhood, i.e. the local branching constraint changes, we apply the bound tightening phase
- Results are still uncertain: no clear winner

## ...and that's all

# Thank you!

# Bibliography

Drud, A. (1985).

Conopt: A grg code for large sparse dynamic nonlinear optimization problems.
*Mathematical Programming*, 31:153–191.

Fischetti, M. and Lodi, A. (2005).

Local branching.
*Mathematical Programming*, 98:23–37.

Fourer, R. and Gay, D. (2002).

*The AMPL Book*.
Duxbury Press, Pacific Grove.

Gill, P. (2006).

*User's guide for SNOPT version 7*.
Systems Optimization Laboratory, Stanford University, California.

Hansen, P. and Mladenović, N. (2001).

Variable neighbourhood search: Principles and applications.
*European Journal of Operations Research*, 130:449–467.

Leyffer, S. (1999).

User manual for minlp_bb.
Technical report, University of Dundee, UK.

Sahinidis, N. and Tawarmalani, M. (2005).

*BARON 7.2.5: Global Optimization of Mixed-Integer Nonlinear Programs*, User's Manual.

Westerlund, T. and Pörn, R. (2002).

Solving pseudo-convex mixed integer optimization problems by cutting plane techniques.
*Optimization and Engineering*, 3:235–280.