VINEOPT

Demo version 1.7

Manual

Kaj Holmberg

August 19, 2008

Contents

1 Introduction **2**

2 Main Window **2**

3 Network types **3**

4 Parameters on the command line **4**

5 Keyboard shortcuts **4**

6 Menus **6**

6.1 File 6

6.2 Optimization 7

6.3 User optimization codes 8

6.4 Visualization 9

6.5 Changes 11

6.6 Help 17

7 Right button menu **17**

8 Scripts **18**

9 Data files **18**

10 Optimization parameters **20**

11 List of all shortcuts **20**

VINEOPT: Visual Network Optimization

1 Introduction

VINEOPT is a program for visualization of and optimization in graphs and networks. It is mainly a graphical interface with possibilities to run various optimization codes.

Its main function is to visualize the graph, and enable changing of problem data, see section 6.5. Data can be read from file and saved to file, see section 6.1. New networks can be created from scratch. Solutions can be visualized in the graph, and numerical data (costs, capacities etc) and solutions (flow, node prices etc) displayed, either in the graph or in tables. The picture of the graph can be modified in various ways, as described in section 6.4.

The program is written in Tcl/Tk by Kaj Holmberg. The code has been transferred to C with the program "mktclapp", and compiled to an executable file. The optimization codes are compiled separately.

VINEOPT is started in a terminal window by the command

```
./vineopt
```

The terminal window will work as console, and information will be printed in this window. It is possible to give parameters on the command line, as described in section 4. (A brute force way of killing VINEOPT is by "Control"-C in the terminal window, which is obviously not recommended.)

2 Main Window

As VINEOPT is started, the main window is opened. It contains the following parts. At the top, under the title row, you find the main menu row, containing the following menus.

- **File** (read data/save data/quit program/etc)
- **Optimization**
- **Visualization** (change the appearance of the network)
- **Changes** (change data)
- **Help**

The menus can alternatively be opened by typing "Alt"-f (**F**ile), "Alt"-o (**O**ptimization), "Alt"-v (**V**isualization), "Alt"-c (**C**hanges), and "Alt"-h (**H**elp).

Under the main graphical window a couple of status rows are displayed. The first one shows which change mode you are in (see section 6.5), the problem name (which is the name of the indata file) and the problem size. Finally, this row displays the coordinates of the mouse (in the scale of the network). The right hand part of the row turns red

when the system is busy, which indicates that one should wait until the current activity is ready before starting another one.

The second status row gives brief instructions of how to proceed in the current change mode, and sometimes information about the previous action. The third status row displays network type, and whether or not data has been changed after the last save.

3 Network types

VINEOPT handles different network types, based on different sets of data. One basic problem type is minimal cost network flow problems in a directed graph. This problem type has the following data.

- Link data: For each link: Starting node, ending node, link cost, lower bound for the flow, upper bound for the flow.
- Demand: For each node: Net sink strength (positive for demand, negative for supply).
- Coordinates for each node: x, y.
- Name for each node.

A related problem type is shortest path problems, which requires that a starting node and/or an ending node for the path or flow is specified. (If no choice has been made, the first node is used as starting node, and the last as ending node.)

VINEOPT can handle directed and undirected graphs (but not mixed). Undirected graphs are used for instance for traveling salesman problems.

All network types require starting and ending nodes for each link, and coordinates and names for each node. The following table lists all the possible network types, and the corresponding data.

	Network type	Data
1	Linear minimal cost flow	Link costs and bounds, demand.
2	Shortest path	Link costs.
4	Only topology	-
5	Undirected network	Link costs.
6	Multicommodity flow, single origin-dest	Link costs, capacities, multicom demand.
7	Multicommodity flow, multiple origin-dest	As 6.
8	Uncapacitated network design	Linear and fixed link costs, multicom demand.
9	Capacitated network design	As 6, plus fixed link costs.

We will not go into details of the optimization problems here (other literature is better suited for that purpose), and will therefore not be more specific about the data of the different problems.

4 Parameters on the command line

It is possible to give parameters on the command line when starting VINEOPT, following the syntax

vineopt - k text1 text2

where k is an integer (or the letter A, B or C) and text1 and text2 two optional strings containing file names (including paths). The action taken depends on k in the following manner.

k=A: Simplified mode that only allows minimal cost network flow problems.
k=B: Simplified mode that only allows undirected graphs.
k=0: Short help.
k=1: The problem in file text1 is read and suitable optimization (possibly specified by text2) is done.

If $k \geq 2$, expert mode is entered.

k=2: The problem in file text1 is read.
k=3: As k=1, but in expert mode.
k=4: The problem in file text1 is read and saved in a postscript file.
k=10: Problem names, optimization parameters and optimal objective function values are read from text1. The problems are read, optimized and the objective function values are compared to the given values. (Used for debugging of optimization codes.)

If $k \geq 11$, no graphical window is opened and the program is terminated afterwards.

k=11: As k=3, but without a graphical window.
k=15: The problem in file text1 is read. A script from file text2 is read. The changes in the script are made, and optimization done. (See section 8.)
k=35: As k=4, but without a graphical window.
k=99: All temporary files are erased, after which the program is terminated.

Adding 50 yields simplified drawing of the graph, which is quicker and suitable for very large graphs which otherwise would take a long time to draw.

Adding 100 yields print level 1 (less output).

Adding 200 yields print level 0 (minimal output).

Setting k=A or B has the effect of cleaning out unnecessary items from the menus, and thus making VINEOPT easier to use. Other than that, there is no added functionality. The only way to change this mode is to restart VINEOPT.

For k=1, 3, 11, either text2 is not given, in which case an optimization menu of the possible choices is opened, or text2 specifies the optimization to be done, in the form of an integer representing the optimization problem to be solved. The possible choices are given in section 10.

5 Keyboard shortcuts

The following keyboard shortcuts can be used in certain situations.

Space: OK (instead of Enter/Return).

F4: Quit VINEOPT.

F3: Close active window.

F10: Open File menu.

The "Control"-key together with an arrow (right/left/up/down) moves the whole network in the corresponding direction. The movement is done in larger steps if in addition the "Shift"-key is kept depressed. (Beware of autorepeat, since then the network might not be drawn between the steps, and might jump off the screen.) Moving the mouse with the middle button kept depressed moves the whole network.

The "Control"-key together with the key + or − zooms in or out. "Control"-(gives larger nodes and "Control"-) smaller. ("Default scale" in the "Visualization"-menu retrieves the network, should you loose it.)

At numerical input, a text window is opened, in which normal editing is possible. (See the Tcl/Tk-manual for a detailed description of editing commands available.) One should avoid entering anything other than numbers. Especially Enter/Return does not do what might be expected, and should be avoided altogether. "Tab" moves the cursor to the next entry or button.

The following keyboard shortcuts can be used to choose change mode.

F2: Move node.

F5: Add node.

F6: Remove node.

F7: Change demand.

F8: Add link.

F9: Change link data.

"Control"-a: Determine starting node.

"Control"-z: Determine ending node.

"Control"-o presents an optimization menu, only containing the optimization problems applicable to the current network type. If there is only one possibility, the optimization problem is solved directly. "Control"-q makes a re-optimization of the same type as done most recently, and could be used for "manual" sensitivity analysis. Pressing "Control", "Shift" and left mouse button simultaneously enables doodling (temporary markings in the graphic window, erased when redrawing the network).

Control-x enables changing the font size of all labels in the graph, while Control-Shift-L enables changing the positions of the labels. This is useful when the numbers in the graph are hard to read.

Control-Shift-S saves the current solution temporarily, Control-Shift-R reads the last temporarily saved solution (and replaces the current solution), while Control-Shift-A reads the last temporarily saved solution and adds it to the current solution.

Do not use Enter/Return in VINEOPT. Instead "Space" can be used to press default buttons.

A complete list of all shortcuts in VINEOPT is given in section 11.

6 Menus

Below follows a more detailed description of the menus.

6.1 File

This menu enables reading indata for a problem from file, and saving problem data on a file, using the current name, or a new name. The keyboard shortcut "Control"-s also saves the problem data. (VINEOPT requires at least one link in order to save a network.) Problem comments are displayed when a problem is read. The problem can be erased (making room for a new network). The picture of the graph can be exported to a postscript file, viewed this way, or printed directly (on a printer with postscript capabilities).

The format of the indata files for VINEOPT is specified in section 9. Different data formats are used for different network types, and the indata file contains information about which network type it contains. The network types are listed in section 3.

The menu also enables saving and reading of a configuration file, which contains information of all colors used for the different items, the positions of the different labels, the sizes of the nodes etc. (VINEOPT is fully customizable.) A script file can also be read and executed, see section 8.

One can list all existing network files, optionally of a specific type, and delete network files.

It is also possible to quickly save the current solution temporarily, read it and possibly add it to the current solution.

Below we describe each entry in the menu.

Open: Read a network from file, replace current network. (Any problem comments are immediately displayed.)

Add: Read a network from file, add to the current network. (Current nodes are renamed.)

New (erase problem): Erase current network. (Used for creating a new network.)

Save: Save current network under current name.

Save As: Save current network under a new name name.

Export graph as postscript: Save a picture of the current network to a file in postscript format.

View graph as postscript: View the current network as a postscript picture.

Print: Print the picture of the current network via postscript format on the default printer.

Temporary read/save solution: Save the current solution (flow or tour) or read the last saved solution (flow or tour). When reading, the solution can be added to the current one or replace it. Only one solution at a time can be saved this way.

Save solution: Save solution (flow, shortest path, tour, links in solution etc).

Read solution: Read ealier saved solution (tour or link set). Add to or replace current solution.

Save configuration: Save configuration (colors, label positions etc) on file.

Read configuration: Read configuration (colors, label positions etc) from file. (This is done automatically when VINEOPT is started.)

Read and execute script: A script (see section 8) is read from file and executed.

Delete network file: Delete a network (i.e. all files containing data for the problem).

List network files: List all existing network files, possibly of a certain type.

System commands: Do various Unix system commands, such as ls.

Quit: Quit VINEOPT.

6.2 Optimization

VINEOPT can be coupled to solvers for many different optimization problems. All of them are run from this menu. The following possibilities exist in the demo version.

The minimal cost network flow problem.
The shortest path problem.
The traveling salesman problem.
The multicommodity network flow problem.
The uncapacitated network design problem.
The capacitated network design problem.

When the optimal solution has been found, VINEOPT will display the total cost to the right in the row above the graphical window. Optionally, a numerical list of all the link flows is shown.

The network type given in the indata file does not uniquely determine which optimization problem that can be solved. Network type 1 could be a minimal cost network flow problem or a directed traveling salesman problem. Network type 5 is in the demo version only an undirected traveling salesman problem. Let us describe each item in the menu, also giving the possible network types for each problem.

Solve traveling salesman problem: Solve the directed or undirected traveling salesman problem, i.e. find the minimal cost tour that visits each node once. Network types 1, 2, 5, 3.

With lpsolve: Solves the following problems by writing an MPS-file and running the code lpsolve. (Not suitable for very hard problems.)

Solve network flow problem (lpsolve): Solve the minimal cost (single commodity) network flow problem. Network type 1.

Solve multicommodity network flow problem (lpsolve): Solve the minimal cost multicommodity network flow problem. Network types 6, 7.

Solve network design problem (lpsolve): Solve the capacitated or uncapacitated network design problem. Network type 8, 9.

With glpsol: Solves the following problems by writing a data file for a model file in GMPL/AMPL-format and running the code glpsol. (Not suitable for very hard problems.)

Solve network flow problem (glpsol): Solve the minimal cost (single commodity) network flow problem. Network type 1.

Solve shortest path problem (glpsol): Solve the shortest path problem. Network type 2.

Solve multicommodity network flow problem (glpsol): Solve the minimal cost multicommodity network flow problem. Network types 6, 7.

Solve network design problem (glpsol): Solve the capacitated or uncapacitated network design problem. Network type 8, 9.

User optmization codes: Run an optimization code supplied by the user. See section 6.3 for details. The code should give either a flow or a tour.

Set solution to zero: Set the solution equal to zero.

Erase tour: Make current tour empty.

Check present optimization codes: Check if all optimization codes are present. Output on console.

6.3 User optimization codes

Users can run optimization codes of their own design within the framework of VINEOPT, and take advantage of the graphical features of VINEOPT. This facility is mainly intended for testing heuristic ideas for traveling salesman problems, chinese postman problems, rural postman problems and other combinatorial graph problems that can be modeled by undirected networks with link costs, i.e. network types 5 and 3.

Two types of solutions can be handled, either tours, represented by a sequence of nodes, or link sets, represented by starting and ending node for each link in the set. A tour must be a cycle (not necessarily simple), while a link set may be anything, for example a tree (but it does not have to be connected). If a link set constitutes a cycle, it can be transformed into a tour. Links included in the solutions are considered to have one unit of flow.

There are two types of possible codes, "useropt" which should provide a list of links in the solution, and "usertour" which should provide a list of nodes, which will be interpreted as the order of nodes encountered by following a tour.

The user should compile the codes, name them "useropt1", "useropt2" etc, and "usertour1", "usertour2" etc, and place them in the opt-directory of VINEOPT.

The codes should read an indata file named "usin" provided by VINEOPT, containing the following. On the first row, the number of nodes and links, and on the following, on one row for each link, the starting node, the ending node and the link weight.

The useropt-codes should write the solution in the following format. On the first row, the number of links in the solution, and on the following rows, starting node and ending node of each link in the solution.

The usertour-codes should write the solution in the following format. On the first row, the number of nodes in the tour, followed by each node in the tour (one on each row).

The nodes are identified by their indices, not names, so the user codes will read node indices from the input file and should write node indices on the output file. VINEOPT will then associate node names with the indices.

6.4 Visualization

One can decide how the network should be visualized. Nodes, links, nodes names and all numerical data can be displayed or hidden. (Hiding some numerical data may make it easier to read other numerical data.) Links with flow or contained in the optimal solution can be emphasized.

All data and solutions can be viewed numerically in separate windows, and saved on file or printed. The keyboard shortcut "Control"-f displays current flow, and "Control"-t displays current tour. All colors can be changed and a background picture (gif-format) can be inserted in the graphical window. This menu also enables moving and zooming of the picture of the graph. The following entries are available.

What to show in graph: The following items can be chosen or not chosen:

Show nodes.
Show links.
Show node names.
Show demand.
Show link costs.
Show fixed link costs.
Show link capacities.
Show lower bounds on links.
Show link flow.
Show reduced costs.
Show node prices.
Emphasize links with flow.
Emphasize links in tour.
Emphasize links on shortest path.
Emphasize nodes in tour.
Emphasize links in design.
Draw links outside nodes.
Simplified network drawing (quicker).

See current data: Display the choice from the following sub-menu numerically in a window, which can be saved on file or printed.

See link data: See all link data (costs, bounds).

See demand data: See demand data.

See node coordinates: View list of node coordinates.

See starting and/or ending nodes: See starting and ending node (for shortest path).

See node degree: See the degree of each node.

See current solutions: Display the choice from the following sub-menu numerically in a window, which can be saved on file or printed.

See link flow: See the flow on all links.

See link flow and data: See the flow and all data for all links.

See network design: See the links included in the optimal design.

See node prices: See all node prices.

See reduced costs: See the reduced cost for each link.

See shortest path: See the links in the shortest path.

See tour: See the links in the traveling salesman tour.

Step through tour: Mark the next node in the traveling salesman tour. Repeat (by hitting the spacebar) to step through the tour.

Display flow of a commodity: Show the flow of a commodity. (One can cycle through the commodities.)

Display a node: Used to find a certain node in large networks.

Zoom: Zoom the picture of the network in one of the following ways.
Free zoom
Zoom in
Zoom out
Larger nodes
Smaller nodes

Move: Move the picture of the network in one of the following ways.
Free move
Large move down
Small move down
Large move up
Small move up
Large move right
Small move right
Large move left
Small move left

Default scale: Returns the picture of the network to the original scale.

Redraw: Redraw the picture of the network. (Not needed very often.)

Customize Vineopt: Change link and node labels: Move labels in graph. Change label size. Change label place on arc. Move node names into nodes. Revert to default positions. (These settings are saved in the configuration file.)

Choose background color: Choose color on graph background.

Choose colors: Choose colors on graph background and on all labels in graph. Revert to default colors. (These settings are saved in the configuration file.)

Choose background picture: Choose picture (in gif-format) to use as background to the graph. (No scaling is done.)

6.5 Changes

Here changes of the network and solutions can be made. The change mode can be chosen. The choice is valid until another mode is chosen. The function of VINEOPT is very much determined by the change mode. The following modes exist (with keyboard shortcut given, if there is one).

Move node. (F2)
Add node. (F5)
Delete node. (F6)
Change demand of a node. (F7)
Add link. (F8)
Change link data. (F9)
Delete link.
Move link.
Add a node in the middle of a link. Choose starting node. ("Control"-a) Choose ending node. ("Control"-z) Swap two nodes.

(Adding a node in the middle of a link can be used for cases when parallel links are desired, or, by moving the middle node, instead of curved arcs.)

When choosing a node, the one closest to the cursor when the left button on the mouse is pressed, is chosen. A link is chosen by choosing its starting and ending nodes. (VINEOPT does not handle parallel links.) The following is done in the different modes.

Move node: The node closets to the cursor is chosen when the left mouse button is depressed, and moved as long as the button is held down.

Add node: The new node is place at the cursor when the left mouse button is pressed. The node is given demand zero.

Delete node: The node closest to the cursor is removed when the left mouse button is depressed. All links adjacent to the node are also removed.

Change demand: The node closest to the cursor when pressing the left mouse button is chosen. A small window appears, with the current net demand, where the net demand can be entered.

Add link: The node closest to the cursor when the left mouse button is pressed for the first time is chosen as starting node. The node closest to the cursor when the left mouse button is pressed for the second time is chosen as ending node. In other words, an ending node can only be chosen when a starting node already is chosen.

When starting and ending nodes have been chosen, a menu appears with a suggested link cost (the Euclidean distance between the two nodes multiplied with the distance factor), a suggested link capacity (very large) and lower bound for flow (zero). All these number can be changed, and are saved by clicking on "Save".

Change link data: The link is chosen as above, by first choosing starting node and then choosing ending node. Then data is changed in the appearing link data form.

Delete link: The link is chosen as above, and is (if it exists) removed immediately.

Move link: The link to be moved is chosen as above, by first choosing starting node and then choosing ending node. After this, the new starting node and ending nodes are chosen in the same way. The link is moved, keeping all its data.

Add a node in the middle of a link: A link is chosen as above. The data of the original link is placed on the first of the two new links. The second link is given link cost zero, large capacity and lower bound equal to zero.

Choose starting node: The node closes to the cursor when the left mouse button is pressed is chosen.

Choose ending node: The node closes to the cursor when the left mouse button is pressed is chosen.

Swap two nodes: The two nodes are chosen as above, and their coordiantes are exchanged.

Other changes are also possible. There are some possibilities of manually changing solutions in the form of tours or link sets. Under the heading **Manual change of tour** the following is possible.

In the mode **Insert node at end of tour**, one can build a tour. The chosen node is inserted in the tour directly after the previously chosen node. Since it must be a tour, there must exists a link between the previosuly chosen node and the newly chosen one, and there must also exist a link between the newly chosen node and the first one in the tour. Otherwise, the change is impossible, and nothing is done. (For a complete graph, this will not occur.)

In the mode **Insert node in tour**, one must first choose two nodes, between which to insert a new one, and then choose the node to insert there. In the mode **Swap two nodes in tour**, one chooses two nodes, which will swap places in the tour. In the mode **Remove node from tour**, the chosen node is removed from the tour.

Since the tour is not required to be simple, a node can be visited more than once. In this case, it is not enough to choose a node in the tour. One must also choose which occurence of the node in the tour. For this reason, it is also possible to **Remove node from tour via list** and **Swap two nodes in tour via list**. Here a list on the nodes in the tour, in correct sequence, is opened, so it is possible to choose the correct one. The same actually happens if a node which occurs more than once in the tour is chosen graphically, but then only this node is possible to choose in the list.

All these changes may be impossible, if the graph is not complete, since the links needed for the change may be absent.

Under the heading **Manual change of solution**, one may manually change or build a solution in the form of a link set. Here the solution is not required to be a tour or have any specific structure. Note that it is up to the user to ensure feasibility, if there are additional constraints.

In the mode **Add any link to solution**, one simply chooses a link, and the flow on this link is set equal to one (which means that it is inlcuded in the solution). In the mode **Remove any link from solution**, the flow on the chosen link is set equal to zero. In the mode **Change flow on a link**, the flow on the chosen link can be set to any value.

It is also possible to multiply the flow on all links with constant. Finally, **Make solution into tour** will transform a solution in the form of a link set to a solution in the form of a tour. This does not change the solution, only the way the solution is represented, so the solution must already be a cycle before the change.

One can change problem data by **Tabular change of data**, which brings up a table of link data or demand data, in which the data can be changed. In certain cases, when changing much at the same time, this can be preferrable to chosing one link at a time.

Under the heading **Change network type**, one can change from one network type to another. See section 3 for existing types. Some changes causes loss of data, and there is no "Undo". This can also be done with keyboard shortcuts. For example, "Control"-2 gives network type 2 (shortest path).

Indata files contain information of the network type, and the network type is automatically set when reading a file, so there is no need to set network type before reading data from a file.

There are many other possibilities, such as the following. A directed graph can be made undirected and vice versa. Links can be added in order to get a complete graph. A constant can be added to all link costs. It is also possible to set all link costs equal to the same value, or to random values. The same can be done with all link capacities and lower bounds. There are several possibilities of checking the network data. Demand can be summed up (the sum should usually be equal to zero). One can check if the link costs obey the triangle inequality. The degrees of the nodes can be calculated, and one can find out if the graph is connected. All numerical data can be changed in separate text windows. It is also possible to change the problem comments that have been saved together with the problem.

The change menu contains the following items.

Network change mode: Set the mode for changes in the network data. (The solution will be set to zero.)

Mode: Move node

Mode: Add node

Mode: Delete node

Mode: Change demand

Mode: Add link

Mode: Change link

Mode: Delete link

Mode: Move link

Mode: Add node in the middle of a link

Mode: Choose starting node

Mode: Choose ending node

Mode: Swap two nodes

Manual change of tour: Choose mode for manually changing or building a tour. Sometimes a change is not possible, if the graph is not complete, since the solution must be a tour at all times.

Mode: Insert node at end of tour: The chosen node is inserted at the end of the existing tour.

Mode: Insert node in tour: The third node chosen is inserted between the two nodes first chosen.

Mode: Swap two nodes in tour: The two nodes chosen swap places in the tour.

Mode: Remove node from tour: The chosen node is removed from the tour.

Remove node from tour via list: A list of the tour is given, and the chosen node is removed.

Swap two nodes in tour via list: A list of the tour is given, and the two chosen nodes swap places in the tour.

Manual change of solution: Choose mode for manually changing or building a solution. Here the solution is not required to be a tour or have any specific structure. Note that it is up to the user to ensure feasibility.

Mode: Add any link to solution: The flow on the chosen link is set to one.

Mode: Remove any link from solution: The flow on the chosen link is set to zero.

Mode: Change flow on a link: The flow on the chosen link can be set to any value.

Multiply flow with constant: The flow on all links is multiplied by a constant.

Make solution into tour: Changes the type of solution from a flow (where anything is possible) to a tour.

Tabular change of data: Enables tabular changing of the following data.

Change link data: Tabular change of all link data.

Change demand: Tabular change of the demand.

Give starting and ending nodes: Tabular change of starting and ending nodes.

Change network type: Change or initially set network type.

Type: Only topology: Only the topology of the graph. No costs, bounds or demands. (Type 4.)

Type: Undirected (cost): Undirected graph with link costs. (Type 5.)

Type: Shortest path (cost): Directed graph with link costs. (Type 2.)

Type: Linear min cost flow: Directed graph with link costs, upper and lower bounds on the flow of each link and (single commodity) demand. (Type 1.)

Type: Multicommodity flow, single orig-dest: Directed graph with link costs, upper bounds on the flow of each link and multicommodity demand. Each commodity has a single origin and a single destination. (Type 6.)

Type: Multicommodity flow, multiple orig-dest: Directed graph with link costs, upper bounds on the flow of each link and multicommodity demand. Each commodity may have multiple sources and sinks. (Type 7.)

Type: Uncapacitated network design: Directed graph with linear link costs, fixed link costs and multicommodity demand. Each commodity has a single origin and a single destination. (Type 8.)

Type: Capacitated network design: Directed graph with linear link costs, fixed link costs, upper bounds on the flow of each link and multicommodity demand. Each commodity has a single origin and a single destination. (Type 9.)

Make links undirected: Makes all links undirected (set the network type equal to 5). Only link costs are kept. Other link data is removed.

Make links directed: Make all links directed. Each undirected link is replaced by two anti-parallel links.

Check data or solution: Check problem data or solution in the following ways.

Sum up demand: Sum up the total demand. (The sum should be zero.)

Check triangle inequality: Check if the link costs satisfies the triangle inequality. If not, the least constant that added to the link costs makes them satisfy the triangle inequality is given.

Check node degrees: The degree of each node is calculated.

Check if graph is connected: Checks if the graph is connected or not.

Check if tour is feasible: Checks if the solution is a tour.

Check node degree of solution: Calculates the node degrees of a solution, and warns if all degrees are not even.

Check if solution is connected: Checks if the links with positive flow forms a connected subgraph.

Add/remove many links/nodes: Enables modifications that involve larger sets of links or nodes. The following possibilities are given.

Remove links without flow: Remove all links that have zero flow from the network.

Remove anti-parallel links: In case of two anti-parallel links, remove the one with the highest cost.

Add links to form a complete Euclidean graph: The current graph is kept. Links to form a complete graph are added. Costs of the new links are Euclidean.

Add neutral links to form a complete graph: The current graph is kept. Links to form a complete graph are added. Costs of the new links are zero.

Remove nodes with degree less or equal to one: All nodes with degree zero or one are removed from the network.

Multiple change of data: Uniform change of data for many links or nodes:

Move nodes: Change node coordinates as follows.

Put nodes in a circle: Put all nodes along a circle.

Perturb nodes a little: Make small random movements of each node. (Useful for hiding exact coordinates, or if some nodes have exactly the same coordinates.)

Perturb nodes more: Make larger random movements of each node.

Place nodes randomly: Set the coordinates of each node randomly.

Use a combination of the above: Asks for two weights for the circle coordinates and the random coordinates, and calculates the coordinates for each node as a convex combination of the two cases.

Add a constant to all link costs: A chosen constant is added to all link costs.

Add a constant to all link capacities: A chosen constant is added to all link capacities.

Add a constant to all fixed link costs: A chosen constant is added to all fixed link costs.

Set all link costs to the same value: All link costs are set to a chosen value.

Set all capacities to the same value: All link capacities (upper bounds) are set to a chosen value.

Set all fixed link costs to the same value: All fixed link costs are set to a chosen value.

Set all lower bounds to the same value: The lower bounds for all links are set to a chosen value.

Set link costs to random values: The link costs are set to random values.

Set capacities to random values: The link capacities are set to random values.

Set fixed link costs to random values: The fixed link costs are set to random values.

Renumber nodes from 0: The nodes are renamed by numbering them from 0 upwards.

Renumber nodes from 1: The nodes are renamed by numbering them from 1 upwards.

Add 1 to all node names: The nodes are renamed by adding 1 to all node names. (Use only if node names are numbers.)

Subtract 1 from all node names: The nodes are renamed by subtracting 1 from all node names. (Use only if node names are numbers.)

Change distance factor: When adding a new link, the proposed link cost is equal to the Euclidean distance between the two nodes multiplied with a constant, called distance factor. Here this constant can be changed.

Edit/view problem comments file: A window containing the problem comments is opened, enabling changes in the comments.

Edit/view data files: Direct editing of data files. (To be used with caution.)

Edit/view network data file

Edit/view node name file

Edit/view node coordinate file

Edit/view any text file

6.6 Help

In this menu, one can find shorter help texts in Swedish and in English, as well as the larger manuals in postscript. There is also a list of all keyboard shortcuts. It possible to remove all temporary files and to cancel an input phase. The shortcut "Control"-i also cancels an input phase.

Help text, demo version, in English (postscript)

List of shortcuts

See problem comments

Change user level: User level can be set to "Novice" or "Expert". An "expert" gets less questions of the type "Do you really want to do this?". The shortcut "Control"-e chooses user level "expert".

Change print level: Set print level to minimal (0), less (1), normal (2), more (3) or debug (4), for the console window.

Change directory: Enables changing of default directories. Presently there is only one possibility:

Change data directory: The data directory where VINEOPT first expects to find the network file is changed. This change is saved for future sessions.

Remove all temporary files: Remove all the temporary files that VINEOPT creates. (Useful for cleaning up after a session.)

Cancel input phase: When choosing a link, after choosing the starting node, VINEOPT expects the ending node to be chosen. If no link is to be chosen, this item can be used to cancel the input phase. This can also be done with the shortcut "Control"-i.

Flush logfile: VINEOPT prints much information on the logfile. This menu item makes sure that all text is actually written to the file.

Close logfile: VINEOPT prints much information on the logfile. This menu item closes the logfile, and opens a new one, so that the old one can be inspected without finishing the VINEOPT session.

7 Right button menu

If the right mouse button is pressed when the cursor is in the graphical window, certain changes related to the closest node are enabled. The node can be removed (together with all adjacent links), the name of the node can be changed and the demand of the node can be changed. It is also possible to choose one of the adjacent links, and then change data for this link or delete the link. In this case a small menu appears containing all links adjacent to the node.

8 Scripts

VINEOPT can read simple scripts for data changing, either via the command line of via the "File"-menu. The format of the script is as follows. (This is a more low-level way of changing data, with few error checks, so errors in the script may have unforeseen effects.) Every row should end with ;.

c(i,j)=v;	Set cost for link (i,j) equal to v.
u(i,j)=v;	Set capacity for link (i,j) equal to v.
l(i,j)=v;	Set lower bound for link (i,j) equal to v.
d(i)=v;	Set net demand for node i equal to v.

A few examples:

$c(1,2) = 3;$
$u(3,4) = 5;$
$l(6,7) = 8;$
$d(9) = 10;$

9 Data files

Data manipulation in VINEOPT is mainly supposed to be done in the graphical window, as there are error checks which might help avoiding mistakes. However, sometimes one would like to change data directly in the data file. There are less error checks when reading a data file, so this should be done with care.

Furthermore, sometimes one would like to inspect a data file, to see what is really there. Therefore we here give the format of the data in the data files of VINEOPT. (Assume that the problem name is "example".)

The main network data lies on file "example.net". The first row contains a single zero. The next row contains the network type, 1, 2, 4, 5, 6, 7, 8 and 9.

The third row contains the number of nodes, m, and the number of links, n and, for network types 6, 7, 8, 9, the number of commodities, k. After this follows n rows with link data, one row for each link. The contents of these rows depend on the network type, shown in the following table, where s is the starting node for the link, t the ending node for the link, c the link cost, l the lower bound for the flow on the links, and u the upper bound on the flow on the link.

	Network type	Link data
1	Linear minimal cost network	$s\ t\ c\ l\ u$
2	Shortest path	$s\ t\ c$
4	Only topology	$s\ t$
5	Undirected links	$s\ t\ c$
6	Multicommodity flow, single OD	$s\ t\ c\ u$
7	Multicommodity flow, multiple OD	$s\ t\ c\ u$
8	Uncapacitated network design	$s\ t\ f_0\ c$
9	Capacitated network design	$s\ t\ f_0\ c\ u$

In all types, the origin s and destination t should be node names.

VINEOPT does not handle parallel links, so if there are several rows with the same starting and ending nodes, only the first one will be read. This is also the case for anti-parallel links in undirected networks. (VINEOPT keeps $s < t$ for all links in undirected graphs.)

After the link data follows the demand data, for network types 1, 6, 7, 8 and 9. For network type 1, there are m rows giving the net demand (net sink strength) for each node. (A positive amount for demand, a negative for supply.) The sum of these numbers should be zero.

For network types 6, 8 and 9, there are k rows, one for each commodity, containing the origin for the commodity, o, the destination d, and the demand between these nodes, v. o and d should be node names.

For network type 7, there follows m groups of k rows, one row for each node and each commodity, containing the net demand of the node of the commodity. The net demand should sum up to zero for each commodity.

The coordinates of the nodes are read from a separate file, "example.crd". The first row contains the number of rows, m, and then follows one row for each node, containing x-coordinate and y-coordinate. (If no coordinate file exists, the coordinates will be randomly generated.)

Also the node names are read from a separate file, "example.nam". The first row contains the number of rows, m, and then follows one row for each node, containing the name of the node. (If no node name file exists, the nodes will be numbered consecutively as they appear in the net-file.)

If the file "example.txt" exists, its contents is read as text and saved as problem comments. Here one can for example place information about when, how and by whom the instance was created. The problem comments are shown, when reading a network file.

If the figure is saved on file in postscript format, the file is named "example_1.ps" the first time (i.e. if no file named "example_1.ps" exists). The second time (i.e. if a file named "example_1.ps" exists) the file is named "example_2.ps", and so on (in order not to overwrite existing files). If a text window (with data or solutions) is saved on file, the file name will be "example_1.txt", "example_2.txt" and so on.

Communication between VINEOPT and the solvers is done via internal files. The different optimization codes communicate with VINEOPT via input and output files, and a logfile. These files are not meant to be handled by the user, but are kept to enable checking of input and output data actually used, in case of an error. The input files have names ending with 'in', and the output files have names ending with 'out'. The logfile names all start with 'log'. When the optimization codes are using mps-format, there is also a file named 'mps'. These files are not deleted automatically when finishing a VINEOPT session, but can be deleted via the "Help" menu, or afterwards by giving the command parameter `k=99`.

The file "logfile.txt" contains selected information about all actions done since VINEOPT was started.

As VINEOPT is started, the file "vno_dir.ini" is read (if it exists). It gives the default directories for the optimization codes, the help file, the problem data files and logfiles for the optimization codes. If the last row in the file "vno_dir.ini" contains the name (and path) of an existing image (in gif-format), it is placed as background in the graphical window. (Here maps or other geographical information may be useful.)

10 Optimization parameters

Each optimization problem that can be solved is represented by an integer. This representation is mostly internal, but can be used when giving the parameters k=1, 3, 11 on the command line. The number is then given as text2, see section 4.

The following choices are presently available.

Parameter	Optimization
9	Traveling salesman problem (LKH)
22	Min cost flow problem (mps, lpsolve)
23	Multicommodity network flow (mps, lpsolve)
24	Cap/Uncap network design (mps, lpsolve)
25	Min cost flow problem (GMPL, glpsol)
26	Shortest path problem (GMPL, glpsol)
38	Multicommodity network flow (GMPL, glpsol)
39	Cap/Uncap network design (GMPL, glpsol)
100+i	Usertour i.
200+i	Usersol i.

Each optimization code can only be used for certain network types. The following table gives the possibilites.

Network type	Optimization code
1	22,25
2	9,26
4	-
5	9
6	9,23,38
7	9,23,38
8	9,24,39
9	9,24,39

11 List of all shortcuts

Below follows a list of all shortcuts in VINEOPT. Some of them are only available in certain situations. The keys to be pressed are spelled out. B followed by a number corresponds to a mouse button.

Space: "OK" i.e. press the default button (if there is one).

F1: Display the "about box".

F2: Set “move node” mode.

F3: “Cancel/Dismiss” i.e. close active minor window.

F4: Quit VINEOPT.

F5: Set “add node” mode.

F6: Set “remove node” mode.

F7: Set “change demand” mode.

F8: Set “add link” mode.

F9: Set “change link” mode.

F10: Open “File” menu.

B2-Motion: Move whole graph.

B3: Node menu, see section 7.

B1-Motion: Move node.

Alt-f: Open main menu **F**ile.

Alt-o: Open main menu **O**ptimization.

Alt-v: Open main menu **V**isualization.

Alt-c: Open main menu **C**hanges.

Alt-h: Open main menu **H**elp.

Control-plus: Zoom in.

Control-minus: Zoom out.

Control-asterisk: Free zoom.

Control-parenleft: Increase node radius by one.

Control-parenright: Decrease node radius by one.

Control-Up: Move up.

Control-Down: Move down.

Control-Right: Move to the right.

Control-Left: Move to the left.

Control-Shift-Up: Move more up.

Control-Shift-Down: Move more down.

Control-Shift-Right: Move more to the right.

Control-Shift-Left: Move more to the left.

Control-Shift-equal: Free move.

Control-o: Do optimization.

Control-q: Repeat optimization.

Control-s: Save data.

Control-e: Set user “Expert”.

Control-i: Cancel input.

Control-f: Show flow.

Control-t: Show tour.

Control-x: Change font size.

Control-Shift-L: Change the labels.

Control-Shift-S: Save current solution (temporarily).

Control-Shift-R: Read last (temporarily) saved solution. Replace the current one.

Control-Shift-A: Read last (temporarily) saved solution. Add to the current solution.

Control-a: Set “choose starting node” mode.

Control-z: Set “choose ending node” mode.

Control-Shift-B1: Doodle.

Kaj Holmberg
kahol@mai.liu.se
080808