Chapter 20

Interactive Graphs: network, ibaf, jump

Since 2002 DDLab included two types of interactive graphs: the "network" and "jump" graphs — a third type was added in the 2023 update, the "interactive basin of attraction field"-graph ("ibaf-graph"). The three types, summarised below and in "Interactive Graph Visualization in DDLab"[57], have very similar on-the-fly functionality whereby nodes linked by directed edges can be dragged/dropped and modified together with their defined components/fragments. Many other options were updated and enhanced (section 0.5) including arbitrary text labels in addition to the range of default node displays, so this chapter 20 was largely rewritten, and related revisions were made throughout the book. The 2024 DDLab update includes further minor revisions/inprovements (section 0.6) covered in this chapter.

| _graph type_ ...	_summary_
network-graph 20.5 ...	represents the wiring scheme of a CA, RBN or any discrete dynamical network with n nodes (displayed as discs or numbers) where each node has $k \geq 1$ inputs (its neighborhood, chapter 9) and a variable number of outputs. There may be a k-mix and random wiring. The weight of discs/links can toggle between inputs or outputs. Links can be omitted to run space-time patterns within a network-graph layout (sections 20.20, 32.19, figures 32.42, 32.43) which can also be scrolled (figures 4.10, 20.27).
ibaf-graph 20.6 ...	is an _interactive_ image of the usual basin of attraction field as specified and drawn (chapters 25, 26) with the exhaustive algorithm (section 29.7) so the method works equivalently for null boundaries (section 26.1), random maps (section 29.8) and sequential updating (section 29.9). The ibaf-graph has v^n nodes (displayed as discs, numbers or patterns) — state-space — and is essentially a network-graph but with out-degree=1 and variable in-degree$\geq$0 — pre-images — which sets the weight of discs/links. Just one basin can be isolated while all others are suppressed.
jump-graph 20.7 ...	shows the probability of jumping between basins due to perturbations of attractor states. Nodes/edges are scaled by basin volume/jump probability. The jump-graph is based on either the basin of attraction field "_f-jump_" or the attractor histogram "_h-jump_" (section 31.7.8). For f-jump, edges can be omitted and basins can be drawn at node positions (section 20.14, figures 20.21, 20.23) and their layout loaded back into the ibaf-graph.

(a) circle layout, one node dragged with its 2-step inputs   (b) spiral/rank/unscramble
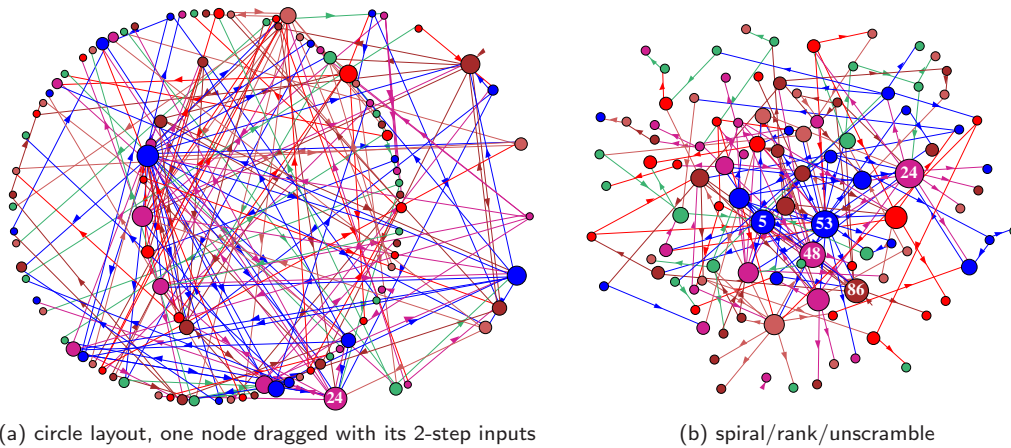
Figure 20.1:   Two versions of the network-graph for the same network with a power-law wiring distribution, both inputs (k=1 to 10) and outputs, n=100. To set up power-law wiring see sections 9.7.2 and 17.9.5. Nodes are scaled according to k. (a) A circle layout, but with one node dragged, together with its 2-step inputs. (b) the same network with spiral layout, then ranked and unscrambled — as in figure 20.12.
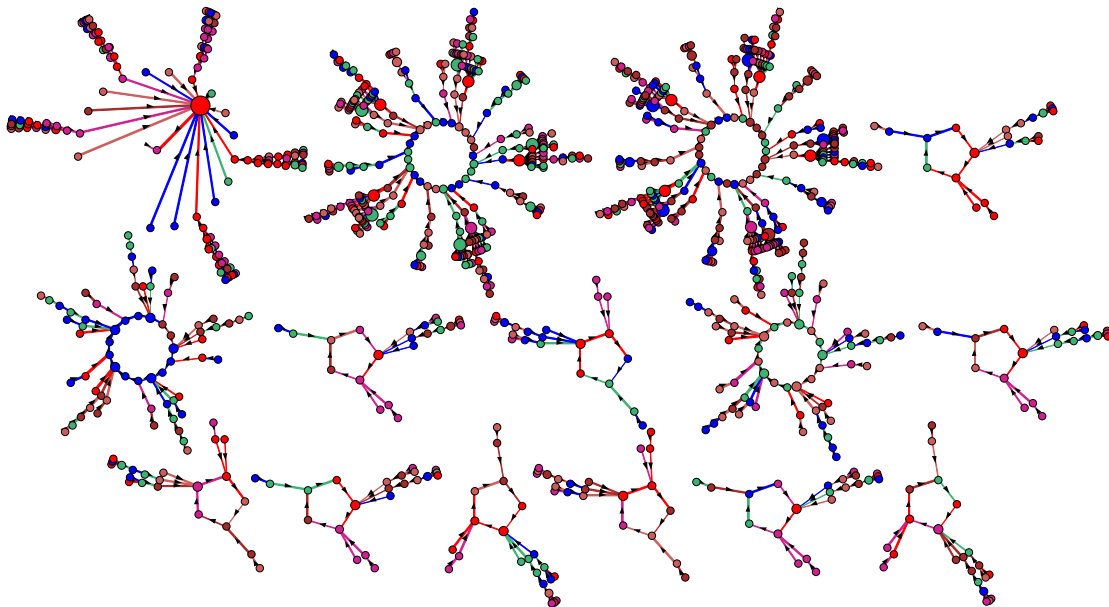

Figure 20.2: The ibaf-graph ($v2k3$, n=10 1d CA rcode 110). Initially nodes are shown as discs and discs/links are scaled by inputs (in-degree). The basin of attraction field is automatically uncompressed (section 26.2) when the ibaf-graph is selected. The layout follows defaults set in chapter 25, but then the basins, independent components of the graph, where dragged to avoid overlaps. The ibaf-graph can be interactively rearranged with the mouse and keyboard for dragging/rescaling/relabeling nodes/fragments/components.
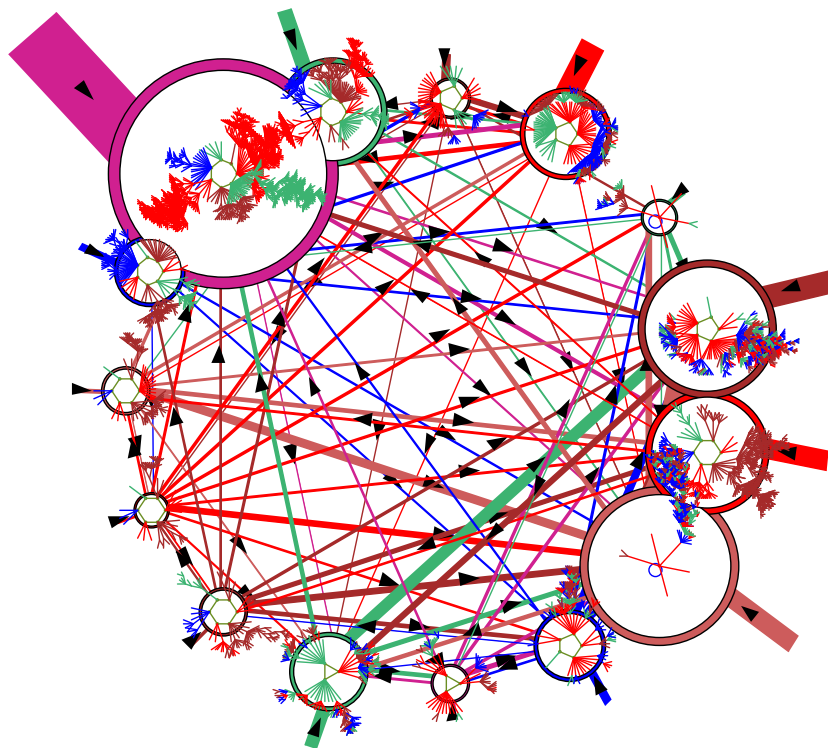
Figure 20.3:   The jump-graph, with basins redrawn within its nodes (RBN $v2k3$, $n=13$, as in figure. 2.5). The jump-graph shows the probability of jumping between basins due to single bit-flips to attractor states. Nodes representing basins are scaled according to the number of states in the basin (basin volume), and can be rearranged and dragged. Links are scaled according to both basin volume and the jump probability. Arrows indicate the direction of jumps. Short stubs are self-jumps. When jump-graph links are eliminated this becomes a layout-graph, providing a method for arbitrarily arranging the basin of attraction field.

20.1   Interactive graphs

The steps for selecting/activating interactive graphs, and the option reminders themselves are covered in sections 20.5 for the network-graph, 20.6 for the ibaf-graph, and 20.7 for the jump-graph. The option reminders differ somewhat between the three types despite a strong overlap.

The graphs and options are presented at two interchangeable stages with distinct top-right reminders: the initial-reminder where options apply to all nodes simultaneously, and the drag-reminder where options generaly apply to a single node, a geometric range of nodes (blocks), or most significantly to the selected node and its linked fragment. Exmples below are for the ibaf-graph — the network-graph and jump-graph reminders are similar.

Once an interactive graph is selected, the initial-graph and its initial-reminder appear first, with options that apply to the whole graph.

initial-reminder applies to whole graph (this example for the ibaf-graph)

IBAF-graph: drag-(def) PScript-P net-# ant-a unscram-u win-w rank0-k
settings-S rot-x/X flip-h/v nodes-(/)/= links-{/} both-[/]
Unreach-U matrix-t/T nodes-n/N links-l Labels-+ arrows-A/</>
layout: file-f graph-g circle/spiral-o/O 1d/2d(tog)/3d-1/2/3 rnd-r/R quit-q:

Overviews of initial-graph options are given in sections 20.2 and 20.3, examples in sections 20.5.3 (network), 20.6.2 (ibaf), 20.7.3 (jump), and the decode list in section 20.9. Enter "**q**" to quit the current graph.

Click the left or right mouse button with the pointer anywhere on the screen (or press **return**) to change to the drag graph and the context dependent drag-reminder (ibaf example below) with options that apply to a selected node (click to activate — initially node 0) and its linked fragment by *inputs*, *outputs*, or *either* (the default), which would include the whole component given no time-step limit (the default), but a time-step distance constraint can be applied (1 to 9).

drag-reminder applies to a node and its fragenent (this example for the ibaf-graph)

node 0, either, step=nolimit: leftb-drag PScript-P elstc/snap-d gap-g
not active?-rightb first, rot-x/X flip-h/v nodes-(/)/=/E links-{/} both-[/] just-j
Lnk0:cut/restore-c/r Lnks0-0:cut/add/restore-C/A/R net-#
step-(1-9) nolimit-0 single-s in/out/either-i/o/e all-a exit-q:

Overviews of the drag-graph are given in sections 20.2 and 20.4, examples in sections 20.6.3 (ibaf) and 20.8 (network/jump), and the decode list in section 20.10. Left/right click on any node to activate it, and drag with the left mouse button depressed. Enter "**q**" to return to the initial-graph and initial-reminder.

20.2   Common themes between initial and drag graphs

By default, nodes (vertices) are displayed as colored discs (with a central number if the disc is big enough) according to the network order, 0 to n-1 for the network-graph, 0 to v^k-1 for the ibaf-graph, and 1 to the number of basins for the jump-graph.

Successive discs are assigned different colors cycling through four colors. Centrally placed black arrows indicate the direction of links (edges). Links are colored according to the parent node and aligned asymmetrically clockwise relative to the parent to separate outgoing and incoming links by a central gap to avoid mutual link overlap. Self-links are show as short stubs projecting from a node. Both the initial and drag graph are displayed in a large central window, and for both, "**PScript-P**" will save the current image as a vector PostScript file (section 20.15), and **net-#** will redrawn the graph, restoring any link cuts or additions made in the drag-graph.

20.2.1   Common presentation options

The following presentation options feature in the both initial and drag graph. They apply to the whole initial-graph, or in the drag-graph just to the active (single) node or the active fragment.

rot-x/X flip-h/v nodes-(/)/= links-{/} both-[/]

*(only "**nodes-(/)/=**" apply for single node status in the drag-graph)*

prompts …	*what they do*
rot-x/X …	lower-case "**x**" rotates clockwise, upper-case "**X**" rotates anti-clockwise.
flip-h/v …	"**h**" flips horizontally, "**v**" flips vertically,
nodes-(/) …	simple brackets: "**(**" to contract, "**)**" to expand the node display as discs, numbers (figure 20.4), or patterns. The font size in the initial-graph will also set the font of disc numbers in both the initial and drag graphs.
nodes-= …	enter "**=**" (equals sign) to cycle through successive node displays (figure 20.5) — 6-way for the ibaf-graph: discs/decimal/hexadecimal/1d-string/2d-string/empty, 3-way for the network-graph and jump-graph: discs/decimal/empty, and 2-way for a graph without links: discs/decimal.
links-{/} …	curly brackets: "**{**" to contract, "**}**" to expand the length of links, or the distance between nodes. Note that scaled link thickness follows disc size.
both-[/] …	square brackets: "**[**" to contract, "**]**" to expand, both nodes and links together.

The opertions for rotations, flips, and contract/expand link length, are made with reference to the window center for the initial-graph, and the pointer position for the drap-graph.
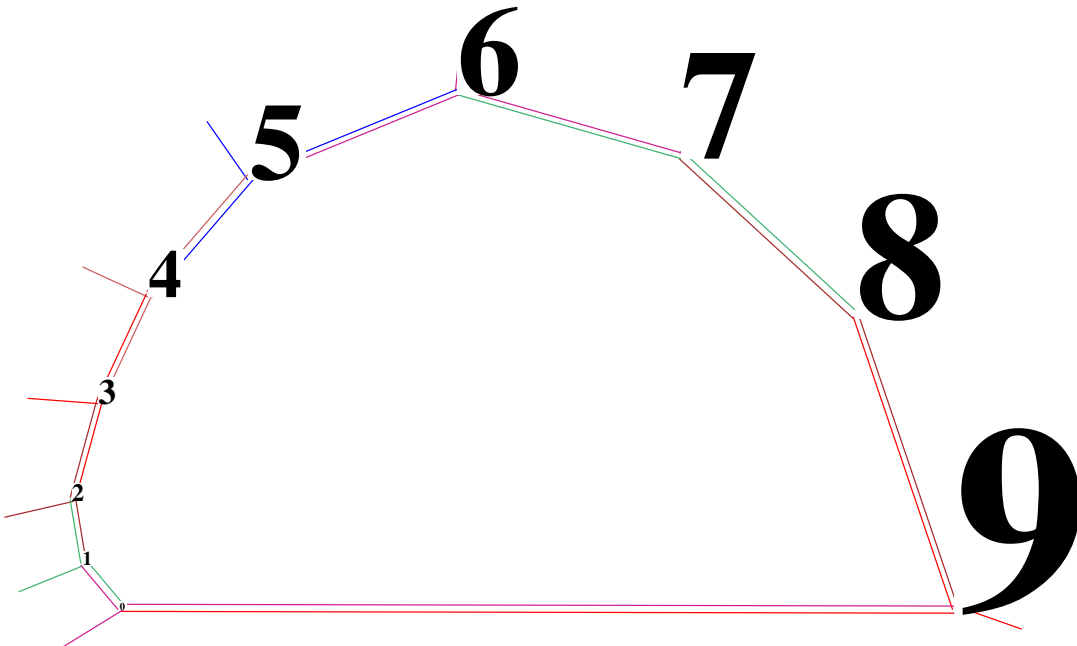

Figure 20.4: A sample from the full range of font pixel sizes from min 6px to max 150px, made with a network-graph in drag mode. Contract/expand with "("/")" by -/+ 1px 6-40, 2px 41-80, 4px 81-150. Note that this does not apply to the DOS version where fonts sizes are constant.
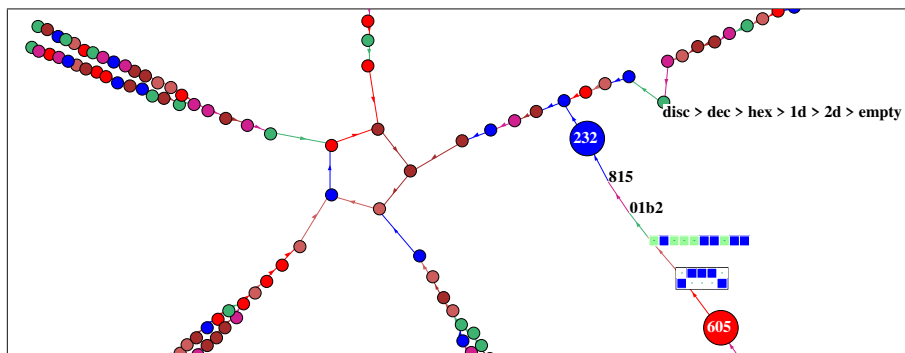
Figure 20.5: Toggling between 6 successive node displays in the ibaf-graph — *Lower Right*: Examples of 5 displays ("empty" is skipped). *Upper Right*: a created "label" (section 20.12) noting the 6 alternatives. A basin for 1d CA $v2k3$, $n=10$, rule 30.

20.3   Initial-graph overview

Options in the initial reminder apply to all nodes simultaneously. Scaled nodes/links is the initial default — the scaling differs by graph type as follows,

graph type title ...	*scaled according to* ...
NET-graph: ...	(section 20.5) ... either inputs (default) or outputs, which can be toggled with "**in/out-z**". The prompt order "**out/in-z**" shows which is active, though this should be evident from the graph — no change indicates that inputs=outputs. "**z**" also turns on scaling nodes/links in case it was off.
IBAF-graph: ...	(section 20.6) ... in-degree (inputs)— the number of a state's (node's) pre-images.
JUMP-graph: ...	(section 20.7) ... basin volume for nodes— the number of states in the basin, links by jump probability — the actual jumps comprising a link divided by the total of possible jumps from the attractor. There are two types of jump-graphs — f-jump from the basin of attraction field, and h-jump from the attractor histogram.

Some significant options unique to the initial-graph are hi-lighted below,

- Nodes can be toggled between scaled/uniform with "**nodes-n**", and links between scaled/thin lines with "**links-l**". The size of link direction arrows can be altered with "**arrows-</>**".

- The pre-programmed graph layouts[1] include "**circle/spiral-o/O**", "**1d/2d(tog)/3d-1/2/3**", and random "**rnd-r/R**". "**2d(tog)**" toggles between square and triangulated layout.

---

[1]These pre-programmed graph layout options would disrupt the ibaf-graph default presentation but are retained because they demonstrate how components can be dragged out of an arbitrary network. The default ibaf-graph layout is restored with "**graph-g**".

- Any layout can be ranked "**rank-k**" by decreasing node weight, unscrambled "**unscram-u**" to separate out weakly connected nodes, and shaken "**rnd-r**" to randomly reposition nodes close to their current positions. The current graph layout can be saved/loaded with "**file-f**".

- Nodes having a given number of inputs or less can be automatically cut (and displaced) in the graph ("**Unreach-U**" section 20.18), which allows leaf nodes to be isolated such as garden-or-Eden states in the ibaf-graph.

- An "ant" can be sent into the graph to trace a path according to link probabilities, keeping track of the frequency of visiting nodes ("**ant-a**" section 20.13).

- The adjacency-matrix of the graph can be generated ("**matrix-t/T**" section 20.19),

- For the f-jump-graph (with or without links), basins of attraction can be drawn at or within nodes ("**basins-i/I/s**" section 20.14).

- The graph window and its previous screen can be toggled with "**win-w**", useful to revisit the basins that were responsible for the ibaf-graph and jump-graph.

Examples of initial reminders are given in sections 20.5.3 (network), 20.6.2 (ibaf), 20.7.3 (jump), and the decode list in section 20.9. Key "**q**" quits the current graph, and a left or right mouse click switches to the drag graph/reminder in section 20.4 below. Most presentation changes in the initial-graph are inherited by the drag graph.

20.4   Drag-graph overview

Options in the drag graph reminder apply to the "active node" (its number is always shown in the title) and a node ensemble if active — the "active fragment" — which is dragged by dragging the active node with the pointer and left mouse button depressed. As well as dragging, the active node+fragment is subject to display/rescale/rotate/flip options (section 20.2.1), so the terms "drag/dragging" also imply all these presentation options. There are four types of interchangeable drag statuses listed below,

| *drag status in title* ...	*functions and selection*
single: ...	(section 20.4.4) to drag a single node — enter "*single*-**s**" at any time. Single node status allows a "block" (below), and also creating multi-line node "labels" (section 20.12). Rotations and flips do not apply to a singe node.
Block x-y: ...	(section 20.4.5) within single node status, enter "*Block*-**B**" to define then drag a geometric block in 1d/2d/3d (section 20.11). The default is set by the last two active nodes. To leave block status and resume "single node" status, enter "*exit-Block*-**B**" or change status with one of the options "**s/i/o/e/a**".
inputs/outputs/either ...	one of these — refered to as "linked fragment" status.
...	to drag a linked fragment (section 20.4.2) — a node and nodes linked to it by inputs, outputs, or either (meaning irrespective of direction) — enter one of the options "*in/out/either*-**i/o/e**" at any time. An unlimited range "**step=nolimit:**", the default, appears in the title (or enter "*nolimit*-**0**") — or set a shorter range by distance measured

in link-steps "**step-(1-9)**" — for example if "**2**" is entered "**step=2:**" appears in the title. Linked fragment status allows link cuts/additions (section 20.4.3).

allnodes: ... enter "**all-a**" at any time to drag a node and the complete graph.

Some issues to note when operating the drag-graph,

- The terms "fragment" or "active fragment" refer to a "block", "linked fragment", or "allnodes". A single node identified as "**single:**" in the title is not a fragment — dragging applies just to that node. However, in all circumstances there is an active node, identified in the title, for example "**node 92**". If a block is active, the active node, whether inside or outside the block, will drag that block along with itself.

- The drag-reminder initially shows "**node 0**" as the active node, but any other node is ready to be activated. To activate a node, click with the pointer at a node-disc center (or the notional center for other node displays) — a right/left click may be required initially. For a node shown as a number, point to the lower-left corner. For a 1d or 2d pattern in the ibaf-graph, point to the upper-left corner. Activation is indicated by the node number appearing in the drag-reminder title and possibly a momentary shake of the graph.

- To drag, click as above but hold down the left mouse button while moving — drag the active node (+ fragment) to a new position which will stabilise on left-button release.

- A graph may comprise nodes with a mixture of node displays, some as decimal numbers (or hex for the ibaf-graph), and also alphanumeric labels, in different font sizes (figure 20.19). While in the actual dragging phase (left button depressed) all alphanumeric displays, including labels, revert to the default font size, or the font size reset in the initial-graph, to optimise animation — once dragging ends (left button released) the correct font sizes are restored.

- For a graph with many nodes (more likely for the ibaf-graph) you will notice that dragging vibrates the graph. This is a consequence of the "elastic" option (the default) which continually redraws nodes/links for graphical animation, but animation/vibration can be suppressed by toggling to "snap" — enter "**d**" to toggle between the two — in the drag-reminder "**elstc/snap-d**" or "**snap/elstc-d**" indicates which drag method is active. With "snap" just the active node is dragged leaving a trail (figure 20.6). On release the active fragment will snap into place. For large graphs this is a more efficient method.

- In linked-fragment status, links can be cut/added according to the link status "**inputs/outputs/either-i/o/e**" with "**Lnk8:cut/restore-c/r**" for the active node, and "**Lnk8-5:cut/add/restore-C/A/R**" between two nodes (section 20.4.2).

- Nodes that have only inputs or only outputs cannot transmitt information — such "gap" nodes can be automatically (and recursively) disconnected with "**gap-g**", which effectvly prunes leaf nodes in the ibaf-graph, starting with garden-of-Eden states (section 20.4.8).

- For the ibaf-graph only, "**just-j**" will toggle to show just the single basin (component) with the active node, to work on a basin in isolation. Enter "**j**" again to restore the complete graph.

- A drag-graph will accept node type/size changes irrespective of the pointer position, but the pointer acts as a reference to expand/contract links or flip/rotate a fragment. Note that flip/rotate is active in fragment status only.
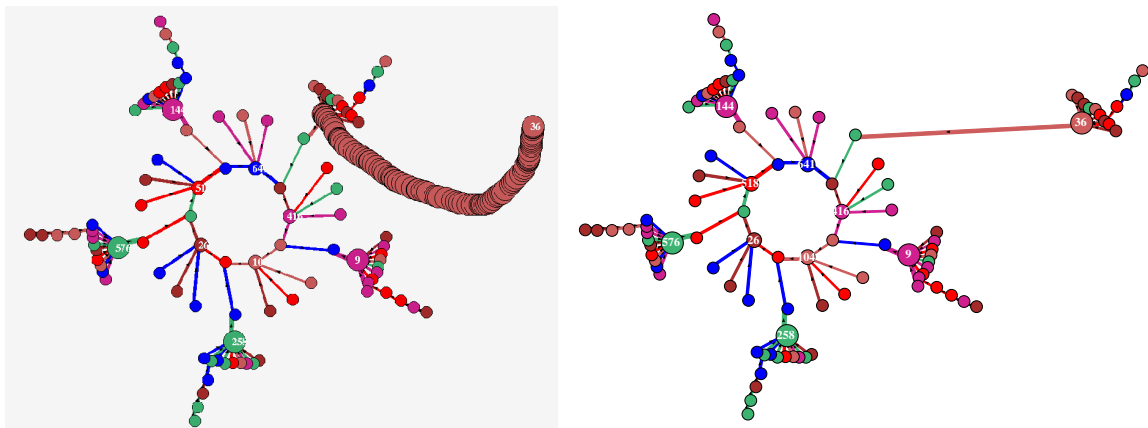
(a) dragging with "snap" leaves a trail   (b) on release the new layout snaps into place

Figure 20.6: Dragging with "snap" (instead of "elastic" — toggle with "**d**"). The ibaf-graph of an isolated basin of attracction — the 4th basin in figure 4.3. (a) the default layout — dragging node 38 to a new position with "snap" active, and the left mouse button depressed. (b) On left button release node 38 and its linked fragment by inputs snaps into place. This is an alternative to dragging with "elastic" animation. (for a 1d CA $v2k3$, n=10, rule 9)

Examples of drag reminders are given in sections 20.6.3 (ibaf) and 20.8 (network/jump), and the decode list is in section 20.10. Key **q** switches back to the initial-graph (section 20.3). Some important drag choices and function are described further below.

20.4.1   drag graph: choices

The choice of options from line 4 of the drag-reminder . . .

> **step-(1-9) nolimit-0 single-s in/out/either-i/o/e all-a exit-q:**

. . . effects drag behavior and the options in line 3. With "**all-a**" set, the whole graph can be dragged irrespective of separate components, and commands apply to all node/links simultaneously, much like the initial-graph. With one of "**in/out/either-i/o/e**" set, just the linked fragment is dragged (section 20.4.2). With "**single-s**" set, just the active node is dragged, but a "Block" can be activated within single node status (section 20.4.5). Line 4 options are described further below.

20.4.2   drag graph: linked fragment in/out/either

To set the link status and create a linked fragment rooted on the active node, apply one of the options "**in/out/either-i/o/e**" where "*in*-**i**" traces inputs upstream, "*out*-**o**" traces outputs downstream, and "*either*-**e**" traces any link irrespective of direction. Enter "*nolimit*-**0**" (zero) for an unlimited distance from the active node. To set a limit in link-steps "**step-(1-9)**" — enter a number between 1 and 9. For the ibaf-graph, link-steps are the same as time-steps in system dynamics.

The title of the drag-reminder shows the active node, current linked fragment type, and step-limit, for example,

node 14, inputs, step=nolimit:
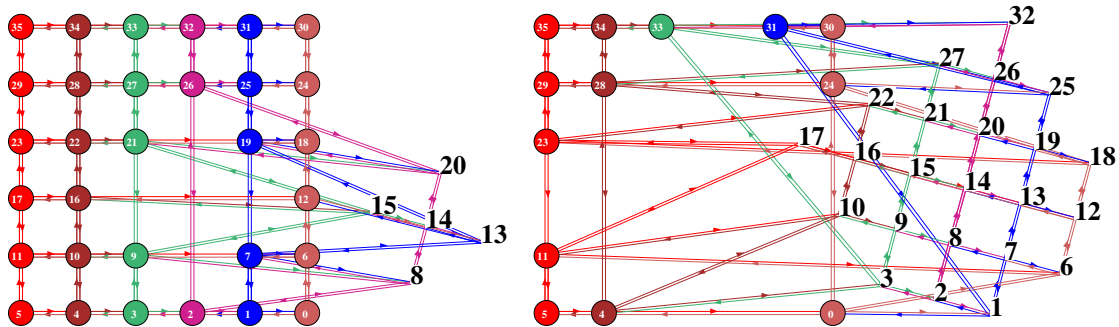**node 14, outputs, step=1:**
node 14, either, step=3:

Figure 20.7: A regular network-graph 6x6x6 with a 2d fragment on active node 14, dragged, rotated, nodes toggled to decimal, and font enlarged. _Left_: Step=1. _Right_: Step=3.
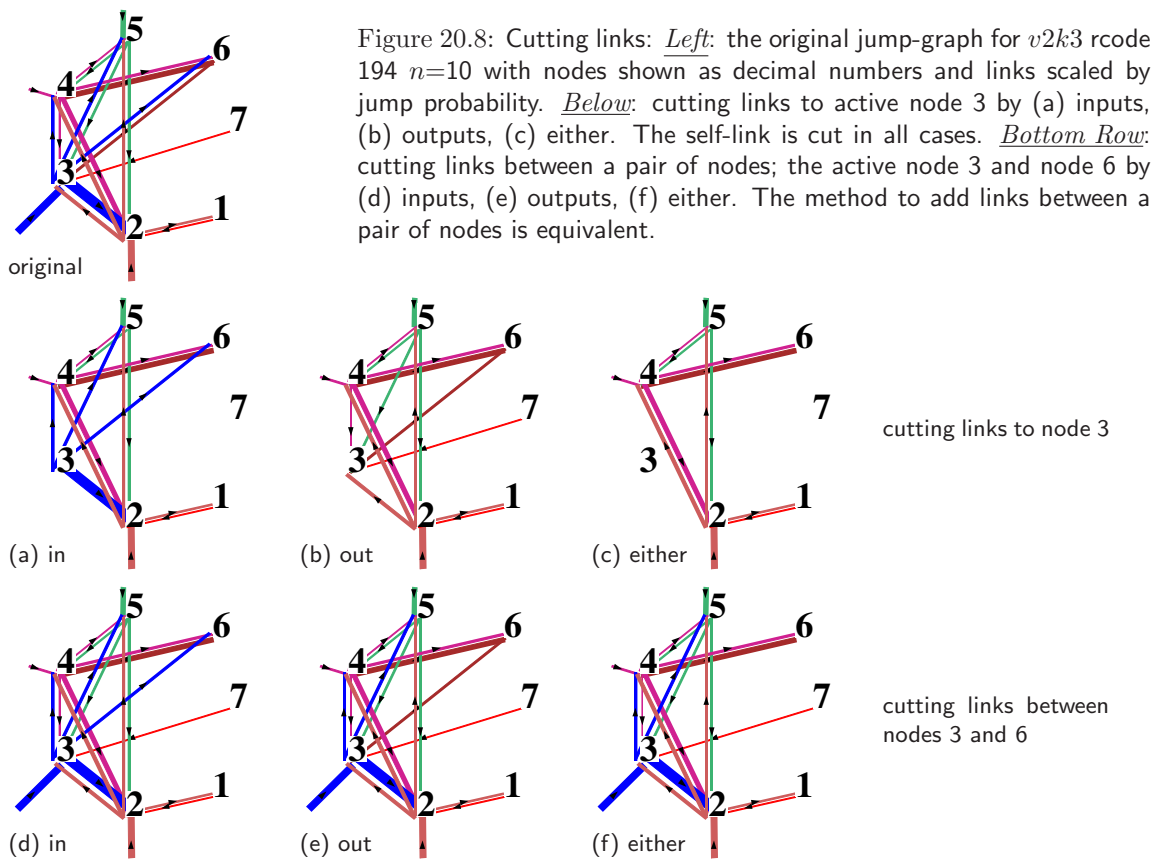
original

Figure 20.8: Cutting links: _Left_: the original jump-graph for $v2k3$ rcode 194 $n=10$ with nodes shown as decimal numbers and links scaled by jump probability. _Below_: cutting links to active node 3 by (a) inputs, (b) outputs, (c) either. The self-link is cut in all cases. _Bottom Row_: cutting links between a pair of nodes; the active node 3 and node 6 by (d) inputs, (e) outputs, (f) either. The method to add links between a pair of nodes is equivalent.

(a) in

(b) out

(c) either

cutting links to node 3

(d) in

(e) out

(f) either

cutting links between nodes 3 and 6

20.4.3   drag graph: cut or add links

The third line of the drag-reminder gives special options to cut or add links, but do not affect the underlying network outside the graph functions[2].

Lnk3:cut/restore-c/r Lnk3-6:cut/add/restore-C/A/R net-# *(for example)*

In options "**cut/restore-c/r**" for the active node **3**, "*cut*-**c**" cuts its immediate links depending on the link status in/out/either (figure 20.8 a — c), "*restore*-**r**" restores the previous cut.

In options "**cut/add/restore-C/A/R**" for the two nodes **3-6** with node **3** active (the defaults are the two most recently activated) "*cut*-**C**" cuts an existing link to node **6** (figure 20.8 d — f), "*add*-**A**" adds a missing link to node **6**, in both cases depending on the link status in/out/either. "*restore*-**R**" restors the original link status previously cut or added. In all cases "*net*-**#**" restores the original network.

20.4.4   drag graph: single node

Enter "*single*-**s**" for single node status (as opposed to a fragment) and click a node — dragging the node and its number appearing in the title of the drag-reminder confirms it is activated,

node 929, single: *(for example)*

While "**single:**" is active, drgging and changes apply to just the active node.

The third line of the drag-reminder gives special options to set a "block" or create a label.

block-B Label-set/tog-L/+ net-#

20.4.5   drag graph: block
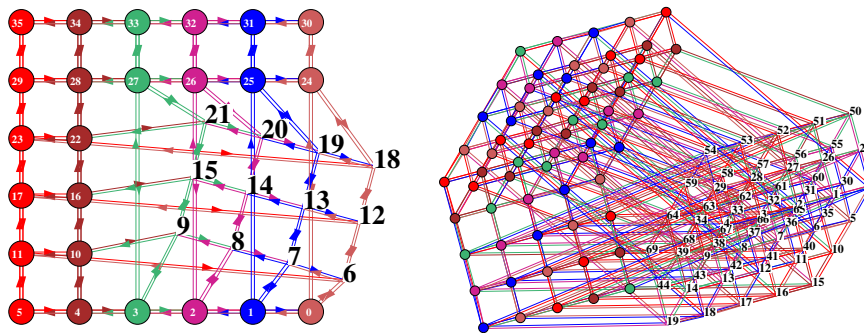
Figure 20.9: A regular network-graph with a block defined, dragged, rotated, nodes toggled to decimal, and font enlarged. *Left*: 2d k=4 6x6 with a 3x3 block. *Right*: 3d k=6 5x5x5 with a 4x4x3 block.

A block is a sequence of node numbers (in 1d, 2d, or 3d) between two selected nodes. In single node status, enter "*Block*-**B**" to open a window to set a block (section 20.11) — the defaults are the two most recently active nodes. An active block will change the title of the drag-reminder,

node 21, Block 6-21: *(for example)*

---

[2]To change the underlying network see chapter 17

While the block is active any changes/manipulations apply to the whole block, and dragging any node (whether inside or outside the block) will also drag the block. The 3rd line of the drag-reminder becomes,

exit-block-B net-#

Enter "*exit-Block*-**B**" to exit the block and revert to single node status, or exit by changing the status with one of "**s/i/o/e/a**".

20.4.6   drag graph: label

In single node status, enter "*Label*-**L**" for a top-right window to create (or remove) a multi-line node label (section 20.12) of up to 90 characters located at the active node. There may be multiple labels at various nodes, in various font sizes, and set at various times, that overwrite the current node displays (figure 20.19), and the labels are inherited when reverting to the initial-graph. Enter *Label*-**+** (the plus symbol) in both drag and initial-graphs, to toggle between showing active labels (throughout the graph) and showing the current node displays.

20.4.7   drag graph: equalise node display

As well as the node display options "**nodes-(/)/=**" (section 20.2.1), in fragment status there is an extra option "**E**" — so "**nodes-(/)/=/E**". Enter "**E**" to "equalise" the node display/size of nodes in the fragment (including **allnodes** if set) according to the active node, which can be outside a "block" as well as inside.

20.4.8   drag graph: gap nodes
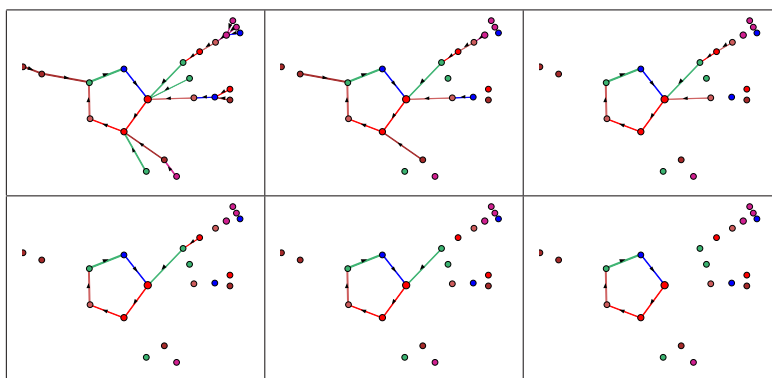
Figure 20.10: Pruning the graph's gap-nodes with "**gap-g**". In this example for the ibaf-graph, one small basin is isolated with "**just-j**" then recursively pruned of its leaf states until only the attractor remains, for a small basin in rcode 110, $n=10$.

In any drag status, enter "*gap*-**g**" to disconnect "gap nodes", cutting off links to nodes which have only inputs or only outputs, so cannot transmit information. Enter "*gap*-**g**" again to cut the next layer of gap nodes and so on, to wittle down the graph into smaller components, finaly to reveal

cycles of directed links. At any stage, pieces of the graph can be dragged. Applies to the whole graph, or to a single component (basin) isolated with "*just*-**j**".

For the ibaf-graph "*gap*-**g**" will initially prune garden-of-Eden nodes, then successive newly created "leaf" nodes until just the attractors remain — try this for a single basin isolated with "*just*-**j**" as in figure 20.10. Enter "'*net*-**#**" to restore the graph.

20.5   The network-graph

The network-graph, an alternative to the "wring graphic" (section 17.3), represents the wiring scheme of a CA, RBN or any discrete dynamical network with n nodes (shown as discs or numbers) where each node has $k \geq 1$ inputs (its neighborhood, chapter 9) and a variable number of outputs. There may be a k-mix and random wiring.

Note that the alternative "wring graphic" has its own distinct and different interactive options to review and amend network architecture (chapter 17), and a new option **graph-g** to generate the network-graph, then toggle between the two, as described in section 20.21.

In the network-graph the weight of discs/links can toggle between inputs and outputs. As with the other graph-types, links can be cut/added (section 20.4.2) allowing any arbitrary network-graph, though the underlying network architecture will not be affected.

With links omitted, space-time patterns can be run within a network-graph layout (sections 20.20, 32.19, figures 32.42, 32.43) which can also be scrolled (figures 4.10, 20.27).

For large networks where only the network itself is of interest, not the rules, it is best to select TFO-mode at the first prompt (section 6.2.1). This allows a mixed-k network (chapter 9) with large max-k, where a power-law k-distribution (section 9.7) may be applied.

The network-graph applies in any mode selected at the first prompt (section 6.1), TFO-mode, SEED-mode, or FIELD-mode, and can be selected at various stages, at the start during the main sequence of prompts, or during a pause/interrupt of attractor basins (section 20.5.1), or when interupting space-time patterns (STP) where as well as all the normal functions of the network-graph, an alternatine STP presentation can be simultaneously run and scrolled within the network-graph (section 20.5.2).

20.5.1   Selecting the network-graph — general case

The network-graph can be activated with "**graph-g**" at the following stages,

- After setting wiring (sections 12.7,17.1).

- From the wiring graphic (section 17.3), which allows toggling between the two representations (section 20.21).

- After the rule or rules have been set (chapters 14 or 16).

- Indirectly from the "attractor basin complete" prompt (section 30.4), — enter "*net*-**n**", or when basins are paused (sections 30.5, 25.3) enter "*ops*-**o**" then "*net*-**n**".

In all these cases, entering "*graph*-**g**" gives this preliminary top-right prompt,

no links -N, show links -def:

Enter **return** to compute and launch the network-graph with links, or "*no links*-**N**" without links. For a larger linked network, this top-right message appears during the computation interval,

'**computing network-graph, 723 nodes, quit-q, or wait...** $\boxed{\textbf{33\%}}$ *(for example)*

	2.	1.	0.	rcode(hex/dec)	
▮ 3	9: 9	9	1	81	129
▮ 2	8: 3	1	6	9e	158
▮ 2	7: 9	0	5	7b	123
▮ 1	6: 4	3	1	1f	31
▮ 3	5: 4	0	7	21	33
▮ 2	4: 3	3	5	77	119
▮ 6	3: 8	3	2	c7	199
▮ 3	2: 8	7	3	47	71
▮ 5	1: 2	1	2	c7	199
▮ 3	0: 5	0	1	c8	200

The network "wiring matrix" (section 17.2) of the RBN $v2k3$, $n=10$, used in some of the examples: the network-graph (figure 20.11), the adjacency-matrix (table 20.1), the jump-graphs (figure 20.23), ant-hits (figure 20.20), and the jump-matrix (table 20.2). (the RBN filename is n10RBN.wrs)

(a) default circle layout, nodes/links scaled by (equal) inputs, $k=3$

(b) nodes/links scaled by outputs — and rearranged

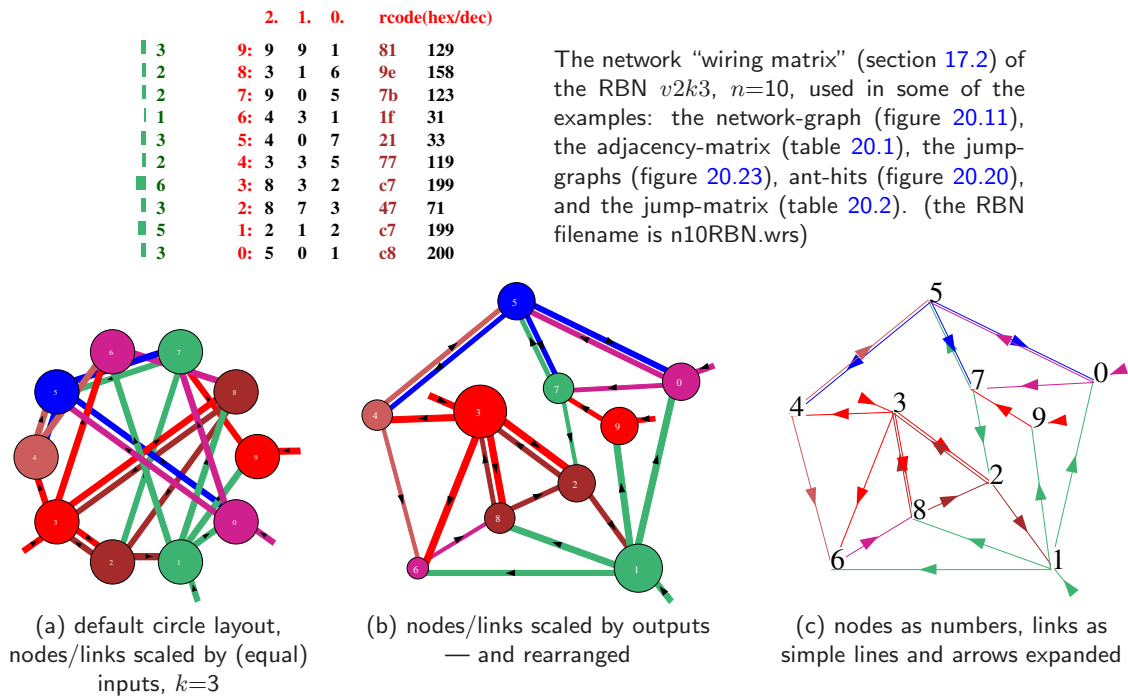(c) nodes as numbers, links as simple lines and arrows expanded

Figure 20.11: Network-graphs of a small RBN (defined *Above*) showing alternative presentations (a) The default circle layout with link width and node diameter scaled by inputs $k=3$, all equal in this example. (b) Layout rearranged by dragging nodes with the mouse, and scaled by outputs. (c) Nodes as numbers and directed links shown as thin lines with arrows.
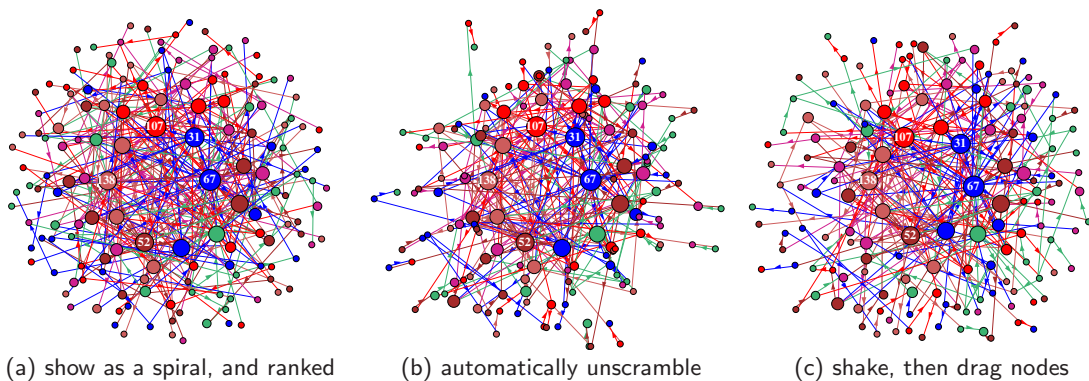

(a) show as a spiral, and ranked

(b) automatically unscramble

(c) shake, then drag nodes

Figure 20.12: One possible method of unscrambling the network-graph of a scale-free RBN, $n=150$. (a): enter "**O**" to show the network as a spiral, then "**k**" to rank the nodes, placing nodes with the highest k near the center. (b): enter "**u**" to automatically unscramble the layout, possibly more than once. (c): enter "**r**" to "shake" the layout, then make final adjustments by dragging single nodes or fragments.

20.5.2   Selecting the network-graph — pausing STP

The network-graph can be activated while running space-time patterns (STP), and then act as a separate template to run parallel simultaneous STP within the network-graph itself.

- While running STP (in 1d, 2d, or 3d) enter "**q**" to pause, giving a top-right window (section 32.16) with multiple context dependent options, but always including "**graph-g**". Enter "*graph*-**g**" for the network-graph.

- A top-right preliminary prompt is presented, where the default "**layout only**" excludes links,

  > **show links -L (caution for large networks), layout only -def:**

  Enter **return** to compute and launch the network-graph without links, or "*show links* **-L**" with links, to run STP simultaneously in the network-graph. Usualy excluding links is preferable. For a larger graph, especially with links, this top-right message appears during the computation interval,

  > **computing network-graph, 723 nodes, quit-q, or wait...** **33%** *(for example)*

- Then the network-graph is shown in a large window occupying most of the screen together with the initial-graph reminder (sections 20.5.3).

- At this point the main space-time patterns are paused and the network graph can be rear-ranged in the usual way with the initial or drag graph reminder (sections 20.5.4). The shape and position can be adjusted as required.

- From the initial-graph reminder, "**q**" will restore normal STP, and start parallel concurrent STP within the latest (unlinked) network-graph layout — if links were selected before they will be automatically excluded. Various on-the-fly options apply to the network-graph STP (section 32.16.5) including diagonal scrolling.

- To diagonally scroll the network-graph STP, the normal STP must first be toggled to 2d with on-the-fly keyhit "**T**", then toggle scrolling on (and off) with key-hit "**#**" — the hash sign. For more details see 'Scrolling the ring" section 4.8.2.

- To pause both the normal STP and network-graph STP (ideally with scrolling off) enter "**q**". The top-right pause options (section 32.16) will reappear, with the option "**graph-g/p**". Enter "**g**" to exit network-graph STP, or "**p**" to save a PostScript snapshot (default filename `my_sgPS.ps`).

20.5.3   The network-graph initial reminder

The network-graph initial-mode reminder appears top-right with options depending on whether or not links are included, The decode is in section 20.9.

> *network-graph initial reminder if links are included*
> **NET-graph:** **drag-(def) PScript-P net-# ant-a unscram-u win-w rank0-k**
> **settings-S rot-x/X flip-h/v nodes-(/)/= links-{/} both-[/]**
> **Unreach-U matrix-t/T nodes-n/N links-l Labels-+ arrows-A/</> in/out-z**
> **layout: file-f circle/spiral-o/O 1d/2d(tog)/3d-1/2/3 rnd-r/R quit-q:**

network-graph initial reminder if links are excluded
NET-graph(nolinks): **drag-(def) PScript- win-w**
settings-S rot-x/X flip-h/v nodes-(/)/= links-{/} both-[/]
nodes-N Labels-+
layout: file-f circle/spiral-o/O 1d/2d(tog)/3d-1/2/3 rnd-r/R quit-q:

20.5.4   The network-graph drag reminder

Enter **return** or a left-mouse click in the initial-graph to switch to the drag-graph and top-right drag-reminder, where the default is "single" node status (section 20.4.4). The network and jump drag-reminders are identical, so both are described in section 20.8.

20.6   The ibaf-graph

The ibaf-graph is an interactive image of the usual basin of attraction field as drawn according to settings in chapters 25 and 26. If the field is redraw within the nodes of a jump-graph (sections 20.14, 24.3.1) the layout can be saved/loaded (a *.grh file — section 20.16) to provide the initial ibaf-graph with arbitrary or special layouts including geometric — circle/spiral/1d/2d/3d.

   The ibaf-graph itself is generated in two stages starting with the exhaustive reverse algorithm (section 29.7) which computes "exhaustive pairs" — the outputs of the entire v^n state-space providing the data for directed links between states. The second stage computes and draws the ibaf-graph with nodes/links scaled by inputs, and nodes depicted as discs (default), numbers or patterns. The exhaustive algorithm limits network size according to table 29.1, but even smaller sizes than those are preferable for practical graphics and computation time of both exhaustive pairs and the ibaf-graph itself — the exhaustive data (*.exh file — section 29.8.1) and the ibaf-graph data (*.grh file — section 20.16) can be saved/loaded to avoid re-computation.

   The ibaf-graph is essentially a network-graph with out-degree=1 and variable in-degree≥0 — pre-images — which provides the weight of discs/links. Just one basin can be isolated while all others are suppressed.

   As well as CA, RBN amd DDN, the exhaustive algorithm makes the ibaf-graph valid for null boundaries (section 26.1), random maps[3] (section 29.8) and sequential updating (section 29.9).

20.6.1   Selecting the ibaf-graph
FIELD-mode only - see also section 24.3 and 29.7.2

A quick-start example for selecting an ibaf-graph starting from the first DDLab prompt is given in section 4.2.3. Here we assume a basin of attraction field has been selected and defined, and we continue as follows,

- At the first **basin parameters** prompt (section 24.1) — enter "*graph*-**g**" to go directly to the jump-graph prompts, or arrive there by viewing the output parameters in sequence. The following top-right prompt is presented (section 24.3),

---

[3]For random maps, to insert basins of attraction into jump-graph nodes, the random map must first be saved (a *.exh file) then re-loaded.

graphs: ibaf-graph-i, jump-graph-j no-edges+L:

Enter "*ibaf-graph-***i**" — "compression" usually on by default for 1d CA, will be turned off automatically with the message,

...**-compression OFF:** *(for 1d CA, enter* **return** *to continue)*

- Enter **return** until the final "basin field" prompt (section 29.4). If "*ibaf-graph-***i**" was entered above, "**ibaf:**" and its valid options appear in blue,

basin field (local) n=10, nonlocal-x ibaf: exh-e rmap-r seq-s nto-o
view ppstack-1: *(valid options for "*ibaf:*" are shown in blue, and underlined here)*

Enter "*exh-***e**" for "exhaustive pairs" (section 29.7.1) — "*rmap-***r**" for a random map or "*seq-***s**" for sequential updating are also valid for their respective ibaf-graphs.

- Then the following prompt is presented,

exhaustive pairs: compute-ret, and save-s, and prt-p/+p:

Enter **return** to compute the exhaustive list (section 29.7 — also describes the other options).

- A red progress bar will appear (section 29.7.1), lasting a brief moment for a small state-space, but taking an exponentially longer time for larger state-spaces, thus the option "*interrupt-***q**",

setting up exhaustive pairs, interrupt-q:

- Then the basin of attraction field is drawn as usual in the main window — a green bar monitors progress (section 30.1). Once finished, a top-right data window and a prompt below the completed bar appear, to select the ibaf-graph (section 29.7.2),

draw interactive basin-of-attraction field graph, ibaf-i:

Entering **return** will skip the ibaf-graph and draw the next basin of attraction field according to mutation settings in chapter 28.
Enter "*ibaf-***i**" to compute and activate the ibaf-graph which is tracked showing the percentage computed so far — enter "**q**" to abandon if the wait is too long,

computing ibaf-graph, 2048 nodes, quit-q, or wait... 23% *(for example)*

- For $n=10$ the elapsed time to compute the ibaf-graph is typically about two seconds[4] and is independent of k or the wiring/rule architecture, but very sensitive to the size of state-space v^n — for binary systems, the time roughly quadruples for each increase of n, a dramatic exponential increase. Its advisable to limit n for this reason, but also because large n results in an unmanageable ibaf-graph.

---

[4]For binary state-space=2^n, approximate elapsed times to compute the ibaf-graph for increasing n, 10:2sec, 11:12sec, 12:32sec, 13:2min20sec. 14:11min, 15:50min (Dell XPS13 laptop).

20.6.2 The ibaf-graph initial reminder

Once computed, the ibaf-graph will be drawn according to the default presentation as in figure 20.13 below, or according to settings in chapters 25 and 26.
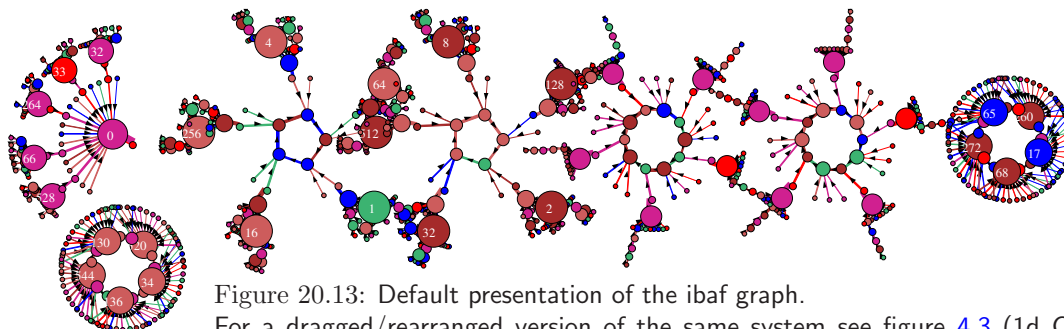

Figure 20.13: Default presentation of the ibaf graph.
For a dragged/rearranged version of the same system see figure 4.3 (1d CA $v2k3$ n=10 rcode 9).

At the same time, the following top-right initial reminder is presented,

> **IBAF-graph: drag-(def) PScript-P net=# ant-a unscram-u win-w rank0-k**
> **settings-S rot-x/X flip-h/v nodes-(/)= links-{/} both-[/]**
> **Unreach-U matrix-t/T nodes-n/N links-l Labels-+ arrows-A/</>**
> **layout:file-f graph-g circle/spiral-o/O 1d/2d(tog)/3d-1/2/3 rnd-r/R quit-q:**

Basins of attraction are independent components and pre-separated in the default presentation, but this would be disrupted by options "**unscram-u**", "**rank0-k**", and non-separated layouts "**circle/spiral-o/O**", "**1d/2d(tog)/3d-1/2/3**", and "**rnd-r/R**". However these options are retained because they demonstrate how components can be dragged out from non-separated layouts. The default separated layout can be restored at any time with "**graph-g**".

Note that the ibag-graph must include links — there are no "layout only" options (without links) as with the network and jump graphs.

20.6.3 The ibaf-graph drag reminder

Enter **return** or a left-mouse click in the initial-graph to switch to the drag graph and top-right reminder below. Left/right click to activate a node — the default is a "linked fragment" indicated by "**either, step=nolimit:**", which means that dragging the current active node will also drag its linked component — the entire basin of attraction. The decode list is in section 20.10.

> *drag-reminder applies to a node and its fragenent, in this case the entire basin*
> **node 230, either, step=nolimit: leftb-drag PScript-P elstc/snap-d gap-g**
> **not active?-rightb first, rot-x/X flip-h/v nodes-(/)/=/E links-{/} both-[/] just-j**
> **Lnk58:cut/restore-c/r Lnks584-640:cut/add/restore-C/A/R net-#**
> **step-(1-9) nolimit-0 single-s in/out/either-i/o/e all-a exit-q:**

The type of linked fragment is changed with "**in/out/either-i/o/e**", and the time-step distance "**step-(1-9)**". Links can also be cut/added. Enter "*single*-**s**" to change to single node status, which allows labels and blocks (sections 20.4.5, 20.11) though blocks are not so useful for the ibaf-graph.

"single" ibaf-graph drag reminder with links, for example

node 144, single: leftb-drag PScript-P elstc/snap-d gap-g
not active?-rightb first, nodes-(/)/= just-j
block-B Label-L/+ net-#
step-(1-9) nolimit-0 single-s in/out/either-i/o/e all-a exit-q:

A special ibaf-graph option is the toggle "**just-j**" to show and work on just the basin with the active node — the other basins are still there but invisible, so might be reactivated by mistake.

20.7   The jump-graph

Perturbations due to noise or external signals are most likely to take affect once a system has relaxed to its attractor because that is where the dynamics spends the most time. Taking single-bit or single-value perturbations to attractor states as the simplest case, the jump-graph represents the probabilities of jumping between basins, and gives some insight into the stability and adaptability of the dynamics, which is especially relevant in attractor models of memory in neural networks and of cell differentiation in genetic networks.

Two types of jump-graph can be created, the *f-jump*-graph — directly from the basin of attraction field (section 20.7.1), or *h-jump*-graph from the "attractor histogram" derived statistically from space-time patterns (section 20.7.2).

To compute jump-graphs, algorithms track where all possible single-bit/value flips to attractor states end up, whether to the same or to which other attractor. The information is presented as a graph with weighed vertices and edges showing jump probabilities, and the data is also available in the adjacency-matrix or jump-table (section 20.19).

The nodes in a jump-graph are numbered starting from node 1 as opposed to nodes in the network or ibaf graphs where the start is node 0 (zero). For the f-jump-graph the number and order of nodes follows the basin of attraction field. For h-jump-graph the number of nodes and order follows attractor types discovered so far (bars in the attractor histogram) and the order of discovery, but the order can sorted by frequency of each type which approximates basin volume, by attractor period, or by average transient length for alternative h-jump-graph orders.

Some differences between the two jump-graph types are worth noting. The f-jump-graph allows basins to be redrawn at nodes, and there are options for "layout-only" without links, either uncompressed, or compressed for 1d CA. The h-jump-graph depends on attractors discovered so far (there may be more as the sample grows) but allows much larger systems, though large systems should be biased towards order, otherwise transients will be too long for the effective accumulation of data. Otherwise the two jump-graphs and their options are very similar.

20.7.1   Selecting the f-jump-graph
FIELD-mode only - see also section 24.3 and 29.7.2

The f-jump-graph applies to any system where a basin of attraction field can be generated with a suitable reverse algorithm, including CA (for 1d with or without compression), RBN or DDN, synchronous or sequential updating, and random maps.

To select the f-jump-graph, proceed as follows,

- Set up a basin of attraction field — "quick start examples" are provides for CA (section 4.2) and for RBN/DDN (section 4.12).

- At the first **basin parameters** FIELD prompt (section 24.1) enter *graphs*-**g** to go directly to the "**graphs:**" prompts, or arrive there by viewing the output parameters in sequence.

- The following top-right prompt is presented (section 24.3),

    graphs: ibaf-graph-i, jump-graph-j no-edges+L:

- Enter "*jump-graph*-**j**" to show the jump-graph with edges — "compression", usually on by default for 1d CA, will be turned off automatically with the message,

    . . .-**compression OFF:** *(for 1d CA, enter* **return** *to continue)*

- Enter "**jL**" for "layout only" without edges. 1d CA compression can be off or on, so the following prompt allows compression to toggle on/off as equired,

    . . .-**compression ON: tog-t:** *(or* -**compression OFF: tog-t:***)*

    Enter "**t**" to switch, or **return** to continue.

- Then enter **return** for defaults at the next 3 prompts, or adjust as required for "mutation" (chapter 28) and "final options for attractor basins" (section 29).

- After the basin of attraction is drawn (chapter 30), the f-jump-graph itself will be drawn in a large central window. If the system is chaotic — a basin of attraction field with many long period attractors — f-jump-graphs (with edges) may take some time to compute. The following top-right message may be temporally displayed, with an inset monitoring progress,

    computing jump-graph, 25 basins, quit-q, or wait . . . **44%** *(for example)*

- To see a sequence of mutant basin of attraction fields and their jump-graphs — enter "**q**" to exit followed by **return**. The required mutation is selected in chapter 28.

20.7.2   Selecting the h-jump-graph
SEED-mode or TFO-made only - see also sections 31.7, 31.7.8

The h-jump applies to any system running space-time patterns.

The attractor histogram (section 31.7) gathers data on attractors and their relative sizes statistically, by automatically running space-time patterns from many random initial states, identifying the different attractors types, and continuously building a histogram of the frequency of falling into each type which approximates the relative basin size. The method, which works in both SEED-mode and TFO-mode, can be applied to large networks, especially RBN/DDN — too large to generate the basin of attraction field due to network size constraints (section 1.6.1).

The h-jump-graph is generated while pausing the attractor histogram. Space-time patterns and the attractor histogram must have been selected. The steps are as follows,

- Select SEED-mode or TFO-mode (section 6.2). Then set up the CA, RBN or DDN.

- Enter **return** at the seed prompt (section 21.1). The initial state is of no consequence as random seeds will be set automatically to generate the histogram.

- In SEED-mode, at the top-right output parameter prompt for **basin parameters** (section 24.1),

    . . . **space-time patterns only-s**, enter "**s**".

In TFO-mode this prompt does not appear.

- At the next top-right prompt for **space-time patterns** (section (31.1),

$$\dots \text{ \textbf{attractors-a}, enter ``\textbf{a}''.}$$

- Enter **return** at the two next top-right prompts for other histogram options (section 31.7),

- Space-time patterns will run top-left from successive random initial states identifying each attractor — each new type is added as a bar to the histogram (x-axis), or added to the bar height/frequency (y-axis). As the histogram builds, rescaling automatically in a lower center window, (section 31.7.2) — information is updated simultaneously top-right and at the top of the of the histogram window (section 31.7.2.1),
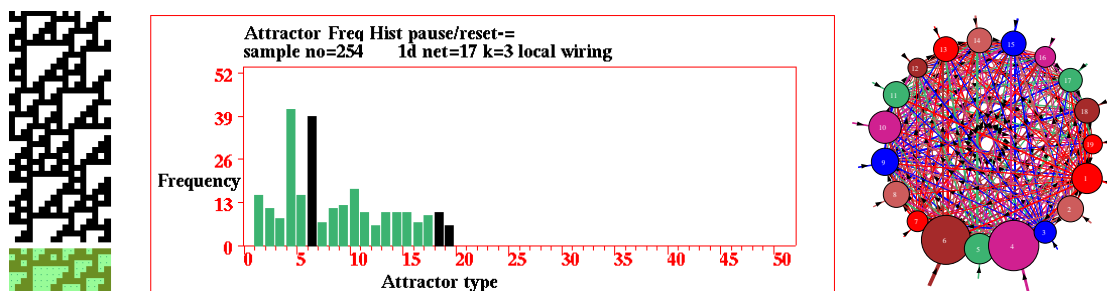

Figure 20.14: A sample of 254 random initial states captures 19 basins. _Left_: STP+attractor, _Center_: unsorted attractor histogram, _Right_: h-jump-graph. 1d CA rcode 110 n=17.

- The reminder "... **pause/reset-=**" is present in the histogram window — at any time enter "**=**" (equals sign) to pause/reset (section 31.7.3). giving a top-right prompt,

    attractors histogram: hist-h stp-q, data:show-d print-p save-S
    redraw-r sort-s data-d undo-pause-u next-ret jump-graph-j:

- Any of these options can be activated if required ("_sort_-**s**" is recommended), before entering "_jump-graph_-**j**" to draw the h-jump-graph (section 31.7.8) in a central window — this top-right message will show temporarily,

    computing jump-graph, 20 basins, quit-q, or wait... 35% _(for example)_

  Then the h-jump-graph will be drawn in a large central window and the initial reminder will show top-right.

- Enter "_quit_-**q**" to quit the h-jump-graph/reminder and return to the histogram, then enter "_undo-pause_-**u**" to continue hunting attractors from the last "... **pause/reset-=**", which can be done repeatedly.

- A larger sample can capture more attractors — toggle space-time patterns off with on-th-fly with keyhit "**S**" (section 32.2) to drastically increase histogram sample speed. To reorder bars by frequency (representing basin volume), during the "... **pause/reset-=**" enter "_sort_-**s**" (section 31.7.7).
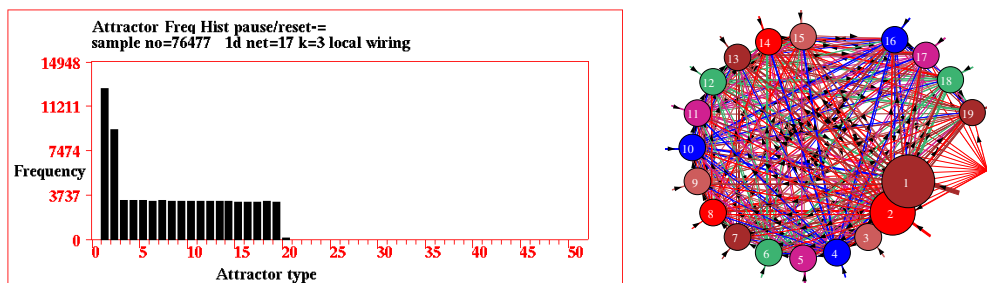
Figure 20.15: A sample of 76477 random initial states captures 20 basins. *Left*: attractor histogram sorted by frequency, *Right*: h-jump-graph with block 16 to 20 dragged. The same CA (rcode 110 n=17) as figure 20.14.

20.7.3 The jump-graph initial reminder

At the same time as the jump-graph is drawn the top-right initial reminder is presented, as below. The h-jump-graph has the title "**JUMP-graph(hist):**" and the options "**basins-i/I/s**" for drawing basins at nodes are omitted.

> *jump-graph initial reminder if links are included,* **basins-i/I/s** *for f-jump only*
> **JUMP-graph: drag-(def) PScript-P net-# ant-a unscram-u win-w rank0-k**
> **settings-S rot-x/X flip-h/v nodes-(/)/= links-{/} both-[/]**
> **basins-i/I/s Unreach-U matrix-t/T nodes-n/N links-l Labels-+ arrows-A/</>**
> **layout:file-f circle/spiral-o/O 1d/2d(tog)/3d-1/2/3 rnd-r/R quit-q:**

For the f-jump-graph in "layout only" (no edges) options, "**Unreach-U**", "**matrix-t/T**", and "**arrows-A/</>**" are omitted.

> *f-jump-graph initial reminder if links are excluded*
> **JUMP-graph(nolinks): drag-(def) PScript-P win-w rank0-k**
> **settings-S rotate-x/X flip-h/v nodes-(/)= links-{/} both-[/]**
> **basins-i/I/s nodes-n/N Labels+**
> **layout:file-f circle/spiral-o/O 1d/2d(tog)/3d-1/2/3 rnd-r/R quit-q:**

The f-jump-graph options "**basins-i/I/s**" draw basins at nodes positions "*basins-**i***" or inside nodes "*basins-**I***" — whereas "*basins-**s***" resets the basin scale. A file of the basin coordinates — of each state in state-space — can be saved and then loaded back into the ibaf-graph. See section 20.14 for details.

20.7.4 The jump-graph drag reminder

Enter **return** or a left-mouse click in the initial-graph to switch to the drag-graph and top-right drag-reminder, where the default is "single node" (section 20.4.4). The network and jump drag-reminders are identical, so both are described in section 20.8.
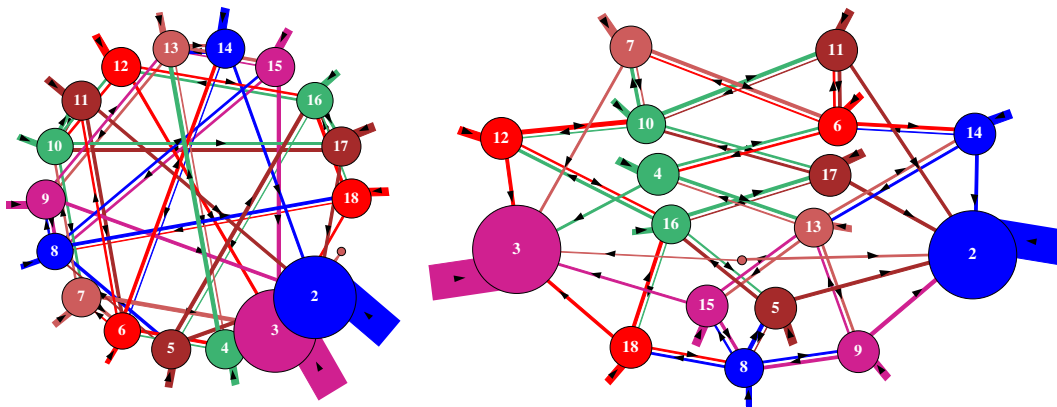
20.8   The jump/network drag reminder

Figure 20.16: The jump-graph for a 1d CA $v2k3$, n=10, rule (dec)141. *Left*: the default circle layout, and *Right*: rearranged with the mouse to clarify the jumps.

The jump and network drag-reminders are identical, so are described together in this section.

Enter **return** or a left-mouse click in the initial jump/network graph to switch to the drag graph and top-right reminder below — the default is single node status indicated by "**node 8, single:**" (for example). Left/right click to activate a node — meaning that just this active node would be dragged oramended, and rotation/flip does not apply. The decode list is in section 20.10.

> *"single" jump/network drag reminder with links, for example*
> **node 14, single: leftb-drag PScript-P elstc/snap-d gap-g**
> **not active?-rightb first, nodes-(/)/=**
> **block-B Label-L/+ net-#**
> **step-(1-9) nolimit-0 single-s in/out/either-i/o/e all-a exit-q:**

Activating a block with "*block-***B**" (section 20.11) or making a label with "*Label-***L**" (section 20.12) is only possible from "single" drag status. After setting a block the prompt changes to show the block range, "**rot-x/X**" and "**flip-h/v**" are included, but "**Label-L/+**" isexcluded,

> *"block" jump/network drag reminder with links, for example*
> **node 2, Block 2-5: leftb-drag PScript-P elstc/snap-d gap-g**
> **not active?-rightb first, rot-x/X flip-h/v nodes-(/)/=/E links-{/} both-[/]**
> **exit-block-B net-#**
> **step-(1-9) nolimit-0 single-s in/out/either-i/o/e all-a exit-q:**

Enter "*exit-block-***B**" or "*single-***s**" to revert to "**single:**" drag status, or enter "*in/out/either-***i/o/e**" for a linked fragment — the prompt changes to show the type of linked fragment and the step limit, and will include options to cut/add links,

"linked fragment" jump/network drag reminder, for example
node 8, either, step=nolimit: leftb-drag PScript-P elstc/snap-d gap-g
not active?-rightb first, nodes-(/)/=/E links-{/} both-[/]
Lnk8:cut/restore-c/r Lnk8-5:cut/add/restore-C/A/R net-#
step-(1-9) nolimit-0 single-s in/out/either-i/o/e all-a exit-q:

If "*all*-**a**" is entered the prompt will show **node 8, allnodes:** (for example) and options to cut/add links will be excluded.

20.8.1   The jump/network drag reminder: no links

Graphs without links can be selected for the network-graph or the f-jump-graph as follows,

network-graph ... with "no-links" section 20.5.1 or "layout only" section 20.5.2.
f-jump-graph ... with "no-edges" section 20.7.1.

The drag reminders without links lack the options relating to a "linked fragment" but drag status for "single", "Block" and "allnodes" are present, examples below,

"single" jump/network drag reminder without links, for example
node 8, single(nolinks): leftb-drag PScript-P elstc/snap-d
not active?-rightb first, nodes-(/)=
block-B Label-L/+
single-s all-a exit-q:

"Block" jump/network drag reminder without links, for example
node 8, Block 3-7: leftb-drag PScript-P elstc/snap-d
not active?-rightb first, rot-x/X flip-h/v nodes-(/)/=/E nodes-(/)= links-{/} both-[/]
eit-Block-B
single-s all-a exit-q:

"allnodes" jump/network drag reminder without links, for example
node 8, allnodes(nolinks): leftb-drag PScript-P elstc/snap-d
not active?-rightb first, rot-x/X flip-h/v nodes-(/)/=/E nodes-(/)= links-{/} both-[/]
line 3 empty
single-s all-a exit-q:

20.9   Initial-graph options

A list of the options in the network/ibaf/jump-graph initial reminders (sections 20.5.3, 20.6.2, 20.7.3) is set out below in roughly the order they appear. Titles in red give the graph type —for example "**NET-graph:**", "**JUMP-graphal (no links)**" — described in "Initial-graph overview" section 20.3. Key-hits usually take effect without entering **return**, but in some cases there are suboptions as noted. Options that require a linked graph are indicated by "(*linked only*)". All initial-graph options apply to the whole graph.

options ...	what they mean
drag-(def) ...	enter **return**, or click the left or right mouse button, for the "drag" options (section 20.10). The drag-reminder will replace the initial-reminder — enter "**q**" to revert. Click on a node to activate it in the drag-graph.
PScript-P ...	to save the current graph image as it appears — a vector PostScript file — suboptions in section 20.15.
net-# ...	(*linked only*) to redraw the graph, restoring any link cuts or additions made in the drag-graph.
ant-a ...	(*linked only*) to launch a projectile — a probabilistic "ant" — in the graph to trace a (Markov chain) path according to link probabilities, keeping track of the frequency of visiting vertices. Top-right suboptions appear — described in section 20.13.
unscram-u ...	(*linked only*) to automatically unscramble the graph, placing weakly connected nodes on the outer edges, nearby their connections. This works best in the circle or spiral layout (figure 20.12). For the ibaf-graph the option is not so useful — it will scramble the basin layout — but can nicely demonstrate dragging components.
win-w ...	to show what lies "below" the graph window, but including the top-right inset restore graph -any key: — so toggling between. For both ibaf and jump graphs, to see the original basin graphic and its top-right data window (section 27.2.1). For the network-graph initiated from the "wiring -graphic" (section 17.3), to compare the two presentations (section 20.21).
rank0-k ...	to toggle between default and ranked node positions. **rank0/rank1** shows which is current. The nodes are ranked (reordered) according to scaled node size, which differs for graph type as follows,

network-graph: (*linked only*) node size based on inputs or outputs, whichever is active.

ibaf-graph: node size based on in-degree (number of pre-images) so inputs. A linear layout (in spiral form in figure 20.17) disrupts the default basin layout, but becomes interesting if ranked because it reflects the "in-degree frequency histogram" (section 24.6).

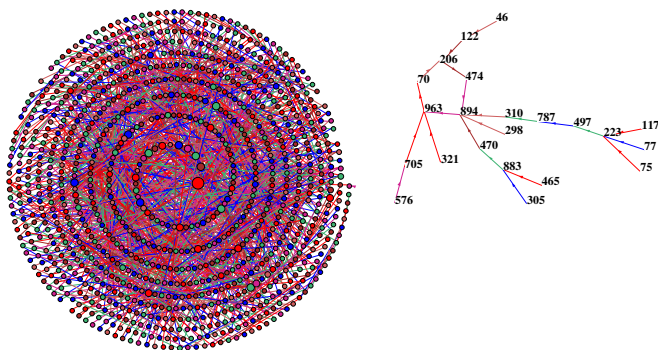**jump-graph:** node size based on basin volume.

Figure 20.17: *Far Left*: Spiral layout of the ibaf-graph in figure 20.22(b) with 2^{10}=1024 nodes, and ranked from the center in descending node size based on in-degree — garden-of-Eden states with in-degree zero form the outer layers. *Near Left*: A component (basin of attraction) dragged out of the spiral, rearranged, with numbered nodes.

settings-S ... to revise the default settings for the rotation angle, and the scaling factors for nodes, links and arrows — suboptions in section 20.17.

rot-x/X ... to rotate the graph about the window center by the default angle of 15 degrees, or a revised angle in "*settings*-**S**" (section 20.17). Enter "**x**" to rotate clockwise, or "**X**" anti-clockwise.

flip-h/v ... to flip the graph about the window center, enter "**h**" to flip horizontally, or "**v**" to flip vertically .

nodes-= ... (equals sign) to cycle (toggle) though successive node displays (figure 20.5) as follows,

ibaf-graph: ... 6-way, discs/dec/hex/1d/2d/empty.

linked network/jump graph: ... 3-way discs/decimal/empty

unlinked network/jump graph: ... 2-way discs/decimal.

nodes-(/) ... (simple brackets) enter "**(**" to contract, or "**)**" to expand the current node display. For discs (and 1d/2d patterns in the ibaf-graph) by the default factor (1.2), or a revised factor in "*settings*-**S**" (section 20.17). For decimal/hexadecimal contract/expand by font pixel size between 6px and 150px (figure 20.4).

links-{/} ... (curly brackets) enter "**{**" to contract, or "**}**" to expand the length of links (or distance between nodes for unlinked graphs) by a default factor (1.1), or a revised factor in "*settings*-**S**" (section 20.17).

both-[/] ... (square brackets) enter "**[**"to contract, or "**]**" to expand both nodes and links/distances at the same time by the default factors or revised factors above.

basins-i/I/s ... (*f-jump-graph only*) enter "*basins*-**i**" to redraw basins at jump-graph nodes but without the graph itself (figure 20.21), or enter "*basins*-**I**" to keep the graph and draw basins inside (or on top of) nodes (figure 20.3, 20.23, 20.24). Enter "*basins*-**s**" to change the basin scale — with suboptions in section 20.14. This is an alternative method for a flexible basin layout with geometric possibilities. Once drawn, basin state-space coordinates can be saved, to reload into the ibaf-graph (section 20.14.1).

Unreach-U ... (*linked only*) to identify and separate unreachable or hard to reach nodes in the graph, (section 20.18).

matrix-t/T ... (*linked only*) to toggle between the graph and its corresponding "jump-table" or "adjacency-matrix" — top right suboptions appear described in section 20.19. For the network-graph and jump-graph, "*matrix*-**t**" shows the numbers of links, "'*matrix*-**T**" the fractions of total links from each node. For the ibaf-graph, "*matrix*-**t**" shows the matrix as an unnumbered pattern (section 20.19.3) and "*matrix*-**T**" shows the list of exhaustive pairs in the terminal as in section 29.7.4.

nodes-n ... (*any graph-type, but network-graph linked only*) to toggle between scaled and unscaled nodes.

nodes-N ... to toggle node numbers on discs. Numbers only appear on discs that are big enough with a diameter ≥ 1.4 of the current text height. Disc number size corresponds to the font size set with "*nodes*-**(/)**" above.

links-l ... (*linked only*) to toggle between scaled links/edges and thin lines (figure 20.11).

Labels-+ ... to toggle between existing "Labels" and the current node display. Labels can be created/deleted in the drag-graph (section 20.12).

arrows-A/</> ... (*linked only*) enter "*arrows*-**A**" to toggle showing arrows. Enter "*arrows*-**<**" to decrease, or "*arrows*-**>**" to increase, the arrow size by 10%. The default arrow start size can be changed in "*settings*-**S**" (section 20.17).

in/out-z ... (*linked network-graph only*) to toggle scaling based on inputs or outputs. "**in/out**" or "**out/in**" shows which is current. Nodes/links will be scaled if curently unscaled.

layout: *options* ...

file-f ... to save node coordinates of the current graph layout, or load an existing file of the same system — a `.grh` file — suboptions in section 35.3.

graph-g ... (*ibaf-graph only*) to reset the default ibaf-graph layout of separate basin components, useful if it was rearranged with the other layout options below.

circle/spiral-o/O ... enter "*circle*-**o**" to show the graph as a circle, or "*spiral*-**O**" as a spiral. Circle layout is the default (except for the ibaf-graph and the 2d or 3d network-graph). Spiral layout requires at least 30 nodes to look right.

1d/2d(tog)/3d-1/2/3 ... enter "**1**", "**2**" or "**3**" to show the graph arranged in 1d, 2d or 3d (examples in figures 10.5 - 10.7). This is especially relevant for a network-graph where the underlying network is 2d or 3d — then the graph in 2d or 3d is the default. The initial 2d network-graph is either square or hexagonal according to the underlying network — thereafter key "*2d(tog)*-**2**" toggles between square and hexagonal (as in figure 10.6).

rnd-r/R ... enter "*rnd*-**r**" to "shake" the layout, repositioning nodes randomly nearby their current position. Enter "*rnd*-**R**" for a completely random layout.

quit-q ... to quit the initial-graph, returning to a point where the graph was selected. For example, the network architecture prompt (section 17.1) for the network-graph, and the "Attractor basin complete prompt" (section 30.4) for the ibaf-graph and f-jump-graph, where **return** would generate the next mutant.

20.10   Drag-graph options

A list of the options in the network/ibaf/jump-graph drag reminders (sections 20.8, 20.6.3) is set out below in roughly the order they appear. The titles in red start with "**node x:**" giving the active node's number followed by the drag status described in "Drag-graph overview" section 20.4.
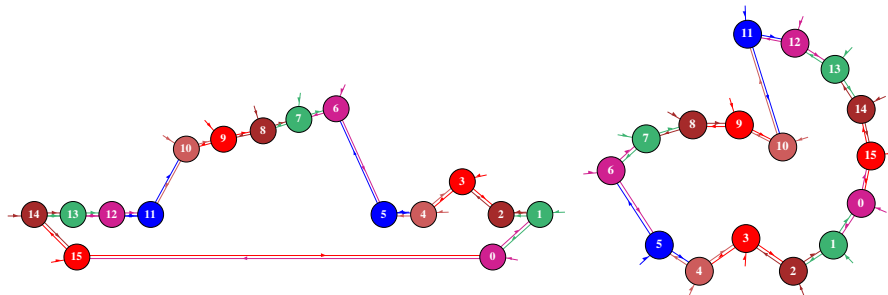
Dragging network-graph nodes and fragments of a 1d CA n=16, k=3, starting with _Left_: a network-graph with 1d layout and _Right_: circle layout. Single nodes have been dragged, and in both cases node 8 and its 2-step neighbors have been dragged and rotated.
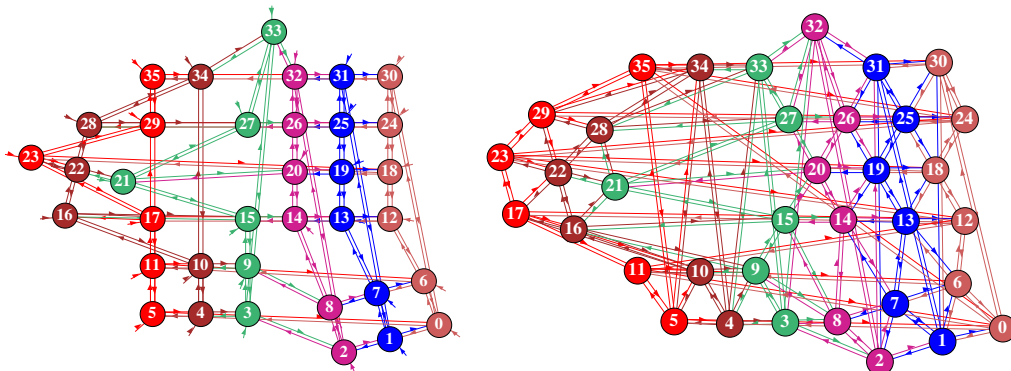
Dragging network-graph nodes and fragments starting with _Left_: a regular 2d network-graph of a 2d CA 6×6, square k=5, and _Right_: hexagonal k=6 network. In both cases, a node has been dragged from the top row, node 22 and its 1-step neighbors have been dragged and rotated, and a 2d block (0 — 8) has been dragged and rotated from the lower right hand corner.
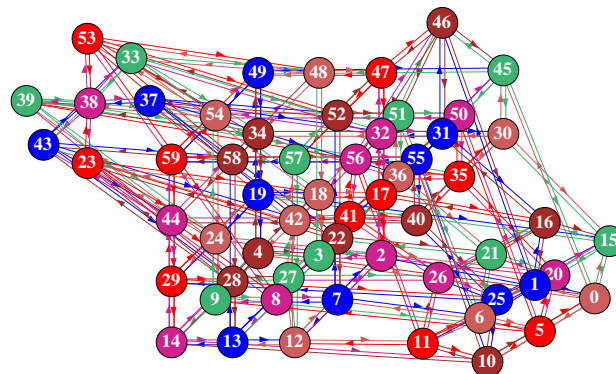
Dragging network-graph nodes and fragments starting with a regular 3d network-graph 5×3×4, k=6. Node 46 was dragged from the top level, node 38 and its 1-step neighbors were dragged and rotated, and a fragment (0 — 26) was dragged and rotated from the lower right hand corner.

Figure 20.18: Examples of dragging nodes and fragments for 1d, 2d, and 3d network-graphs, including single nodes, linked fragments and blocks.

Keyhits usually take effect without entering **return**, but in some cases there are suboptions as noted. The options apply to the active (single) node, or to the active "fragment" which can be a "block", a "linked fragment", or "all nodes" (section 20.4) . Options that require a given context are indicated — for example "(*linked only*)", "(*fragment only*)", "(*single only*)" . Note that the active node can be outside a "block" as well as inside. As a check, drag any node to see which nodes move — these make up the currently active fragment.

options . . .	*what they mean*
drag-leftb . . .	click the left mouse button on a node to activate it and hold down to drag the node or fragment — then release in a new position. Initially, to activate a node it may be necessary to click the right button first, then the left, possibly a few times. Hence the reminder,

<div align="center">

not active?-rightb button-first

</div>

PScript-P . . .	to save the current graph image as it appears — a vector PostScript file — suboptions in section 20.15).
elstc/snap-d . . .	to toggle between two methods of dragging nodes or fragments. By default, a node (+fragment) is dragged with "**elastic**" links and graphical animation Alternatively, animation can be suppressed and the active node dragged leaving a trail — on release, the node (+fragment) and links "**snap**" into place and the trail disappears (figure 20.6). For large graphs "**snap**" is a more efficient method. Enter "**d**" to toggle between the two — the order of "**elstc/snap**" and "**snap/elstc**" shows which is active,
rotate-x/X . . .	(*fragment only*) to rotate the active fragment about any position pointed to by the cursor, by the default angle of 15 degrees, The default can be revised in the initial-graph (section 20.17). Enter "**x**" to rotate clockwise, or "**X**" anti-clockwise.
flip-h/v . . .	(*fragment only*) to flip the active fragment about any position pointed to by the cursor, enter '**h**" to flip horizontally, or "**v**" to flip vertically.
gap-g . . .	(*linked only, applies the whole graph in any status*) to recursively disconnect "gap nodes" which have only inputs or only outputs, so cannot transmit information (section 20.4.8). Applies to the whole graph in any drag status, or to the ibaf-graph as a single basin isolated with "*just*-**j**" (section 20.4.8).
nodes-= . . .	(equals sign) to cycle (toggle) though successive node displays (figure 20.5) as follows,

<div align="center">

ibaf-graph: . . . 6-way, discs/dec/hex/1d/2d/empty.

linked network/jump graph: . . . 3-way discs/decimal/empty

unlinked network/jump graph: . . . 2-way discs/decimal.

</div>

nodes-E . . .	(*fragment status only*) to equalise the display/size of all nodes in the fragment according to the active node, which can be outside a "block" as well as inside.

nodes-(/) ... (simple brackets) enter "(" to contract, or ")" to expand the current node display by default factors[5]. For decimal/hexadecimal, contract/expand by font pixel size between 6px and 150px (figure 20.4).

links-{/} ... (curly brackets) enter "{" to contract, or "}" to expand the length of links (or distance between nodes for unlinked graphs) by the default factor[5].

both-[/] ... (square brackets) enter enter "["to contract, or "]" to expand both nodes and links/distances at the same time by the default factors[5].

just-j ... (*ibaf-graph only*) to show just the basin with the active node, which is a separate component, to work on it in isolation. The other basins (components), are still there but invisible, so clicking in an emply space could switch to a different isolated basin. Enter "*just*-**j**" again to toggle between the isolated and all basins.

block-B ... (*"single" status only*) to define a block — suboptions in section 20.11. The outer edges of the block can be set, or defaults accepted — the last two nodes that were activated. For a linked network-graph in 2d or 3d the outer edges can be the outer corners of a 2d or 3d block — **Block 6-21** (for example) appears in the title.

exit-block-B ... (*in block status*) enter *exit-block*-**B** to exit a block and revert to "single" status, or exit by changing status with "*single*-**s**", "*in/out/either*-**i/o/e**", or "*all*-**a**".

Label-L/+ ... (*"single" status only*) to create (or remove) a multi-line label at the active node — suboptions in section 20.12. Enter "*Label*-+" (plus sign) to toggle between all labels and the current node display. Labels persist if reverting to the initial-graph where all labels can also be toggled with "*Label*-+".

Lnk23: cut/restore-c/r ... (*linked-fragment only, for example*) enter "*cut*-**c** to cut (disconnect) the active node from its immediate links, depending on which option, **inputs**, **outputs** or **either** is active. Enter "*restore*-**r** to undo the cuts.

link 23-19: cut/add/restore-C/A/R ... (*linked fragment only, for example*) The active and previously active node numbers appear in the prompt. Enter *cut*-**C** to cut or *add*-**A** to add a link (or links) between the pairs, or *restore*-**R** to restore — depending on which option, **inputs**, **outputs** or **either** is active.

net-# ... to restore the original links in the graph, undoing any cuts or additions. Existing cuts/additions are conserved when reverting to the initial-graph, but can be restored with the initial option "**net-#**".

step-(1-9) ... enter a number between **1** and **9** to limit a linked fragment by the distance from the active node. measured in link-steps (time-steps for the ibaf-graph) — a continuous chain of directed edges depending on which option — **inputs**, **outputs** or **either** — is active. For example,

---

[5]Default contract/expand factors, for discs (and 1d/2d patterns in the ibaf-graph), and also for links, can be reset with *setting*-**S** in the initial-graph (section 20.17).

enter "**1**" for immediate links, "**2**" up to 2 steps away, etc., up to 9 steps. The current status is shown in the drag reminder, for example **step=1**. For unlimited enter "0" (zero) as below. This can be set in any drag-status but appies for a linked fragment.

nolimit-0 ... enter "*nolimit*-**0**" (zero) for an unlimited linked fragment, a continious chain of links relative to the active node, depending on which option — **inputs**, **outputs** or **either** — is active. Gap nodes would interrupt such a chain. **step=nolimit** is shown in the drag reminder. This can be set in any drag-status but appies for a linked fragment.

single-s ... to set "single" drag status for dragging just the active node. The title changes to **single node 23:** (for example). Single status also allows "blocks" and "labels".

in/out/either-i/o/e ... enter "*in*-**i**", "*out*-**o**" or "*either*-**e**" to define the type of link — **inputs**, **outputs** or **either** (shown in the title) for dragging linked fragments.

all-a ... enter "*all*-**a**" to drag all nodes, the whole graph. The title changes to **all nodes:**.

exit-q ... Enter "**q**" to exit the drag-graph and return to initial-graph and reminder. (sections 20.9). Its easy to flip between the two reminders to implement alternative functions.

20.11   Defining a block
drag-graph — see also section 20.4.5

If "**single:**" status is active in the drag reminder (section 20.4.4) enter *block*-**B** to define a block[6] in 1d, 2d or 3d. For a 2d or 3d network-graph, a block can be defined according to its 2 outer corners. For a jump-graph or a 1d network-graph, the block is simply a range of nodes between an upper and lower limit (as in section 17.6.3). The easiest way to set the block is to click (activate) the outer corners to set the defaults. Make sure these are really activated by checking that the node number appears in the drag reminder, for example "**node 301**". Any pair of opposite corners can be defined by clicking in any order, which will be converted automatically to the lower right and upper left corners in the following top-right prompts, where defaults can be accepted or the block set by hand depending on the type of graph.

> *for a 1d network n=62, or ibaf-graph or jump-graph*
> **define 1d block:**
> **1d block (0-61, def 49-59),** *(values shown are examples)*
> **low:**
> **high:**

---

[6]The block methods are similar in chapter 17 "Reviewing network architecture" (sections 17.6.3, 17.7.5, 17.8.5).

for a 2d network-graph, 22x22
2d network: define block: 1d-1, 2d(def):
2d block (this=5,2 max=21,21), *(values shown are examples)*
low corner (def 5,2) i: j:
high corner (def 7,5) i: j:

for a 3d network-graph, 6x6x6
define 3d network 1d-1 3d-(def):
3d block (this=0,3,0 max=5,5,5), *(values shown are examples)*
low corner (def 0,3,0) i: j: h:
high corner (def 5,3,5) i: j: h:

Enter **return** to accept each default, or enter new coordinates. Enter "**q**" at any point to abandon the block and reveret to "**single:**" status. Once a block is activated,

node 700, Block 267-700: *(for example)*

appears in the title, and "**exit-block-B**" in the drag-reminder. For a network-graph, a 1d block can apply for a 2d or 3d if "**1d-1**" is selected first, otherwise the coordinates of the opposite (low and high) corners define the block.

Once the block is defined, dragging any node (whether inside or outside the block) will also drag the block, and "**rotate-x/X**", "**flip-h/v**", will apply just to the block, centered on the cursor.

Enter "*exit-Block*-**B**" to exit the block and revert to single node status, or exit by changing the status with one of "**s/i/o/e/a**".

20.12   Labels at nodes
drag-graph — see also section 20.4.6

Arbitrary labels of up to 90 characters and multiple lines can be made at graph nodes on top of (or below) the current automatic node display. This is done from the drag-graph in "**single:**" status where "**Label-L/+**" appears in the drag-reminder — enter "**L**" to set a label at the active node. All existing labels are toggled on/off with "**+**" (plus key) in any situation including the initial-graph which inherits labels from the drag-graph. A reminder "**Label-+**" is included in the initial-reminder. Current node displays, with or without labels, appear as usual and are toggled to alternative displays with key "**=**"(equals key).

To set a label make sure "**single:**" status is active, or reset with "*single*-**s**", then click to activate the selected node — top-left "**node 926, single:**" (for example) will appear in the title, then enter "*label*-**L**" for the following top-right label-prompts presented in sequence below — for example, the ibaf-graph in figure 20.19,

node=287 (0-1024) Label: empty *(if a label has not been set, or was deleted)*
accept-ret new-n:
or
node=509 (0-1023) Label: "My special\node is\HERE" *(the current label, for example)*
accept-ret delete-d new-n:

Enter **return** to accept the empty or current label or "*delete*-**d**" to delete a current label. For a new label enter "*new*-**n**" giving further reminders below, together with a flashing cursor on the 4th empty line to enter the new label.

node=287 (0-1024) Label: empty *(for example)*
accept-ret, new-n:n enter new Label max 90 chars *(if "new-n" was selected)*
back-backspace, linebreaks-"\", complete/exit-ret *(reminder notes)*
■ *(flashing cursor for input)*

At the flashing cursor, an immediate "**return**" will also delete an existing label. Type the new label characters (max 90) using the "\" (backslash) key for line breaks and the space bar for spaces. A leading backslash will start text one row down for a numbered node (in dec or hex) to remain visible, as in some labels in figure 20.19. To backtrack use the "**Backspace**" key, not lower case "**q**" which can be part of the label[7], together with most items on the keyboard[8] including punctuation and the space-bar, but "\" (backslash) is reserved for line breaks.

Enter **return** to complete the label and exit the label-prompts. The label appears immediately in the graph with the bottom-left of the first character at the node center if not displaced by linebreaks or spaces. A label will stay attached to its node when nodes or fragments are dragged or the graph rearranged. The current image including labels can be saved as a PostScript file (section 20.15) as usual, but there may be small differnces in text size between the screen and the PostScript images, which can be examined in GhostView (chapter 36).

The default label text size can contract/expand (not for DOS) by applying "**(**" or "**)**" to the active node while in numbered node display — the label font size will correspond, so its a good idea to toggle the node display to numbers prior to setting a label. During the actual dragging phase (left button depressed, drag) all alphanumeric displays — numbered nodes and labels —- revert to the default font size, or the font size reset in the initial-graph — once dragging ends (left button released, drop) the correct font sizes will reappear, though a repeat of drag/drop, just a left click, is sometimes required. Note that "*nodes*-E" to equalise text size in a fragment will include labels.
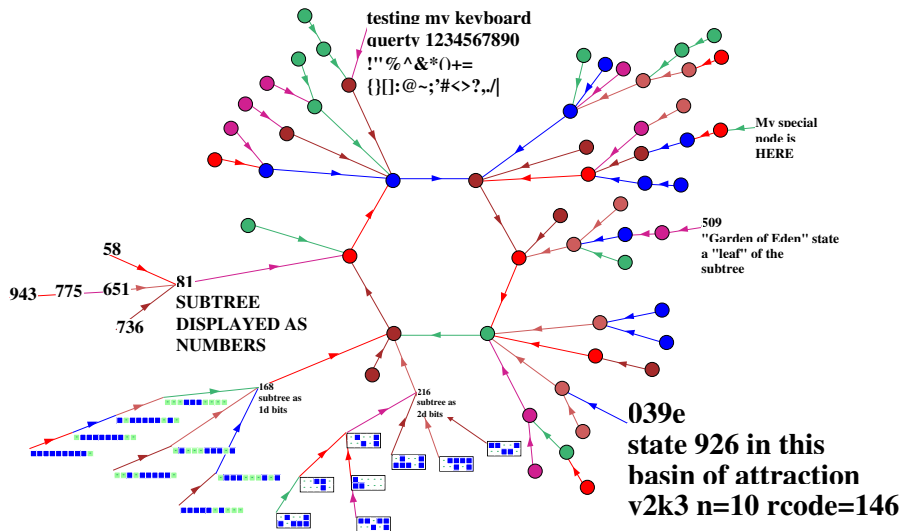
Figure 20.19: An example of arbitrary labels in the ibaf-graph with "*Label*-**L**", in various text sizes. The methods apply equally to the network-graph and jump-graph. In this example the ibaf-graph (v2k3 n=10 rcode 146) has unscaled nodes and edges, and the basin was isolated with "*just*-**j**". The default label starts over its node, but can be lowered with a leading line break to display the node number if active. Nodes can still be toggled (key '=') for single states or fragments.

---

[7]In other contexts in DDLab lower case 'q' is used to backspace, and backtrack in the prompt sequence (section 4.1.2, 4.1.1).
[8]The '£' (UK pound sign) may not work.

20.13   Probabilistic "ant"
initial-graph

Enter "*ant*-**a**" in the initial-graph (section 20.9) to launch a projectile or probabilistic "ant" in the graph. The ant has a red body and leaves a black trail as it moves along the directed links between nodes. The default path starts at a random (or selected) node and traces a continuous Markov chain, a path that targets the next link according to jump probabilities. Alternatively each step can restart at a biased random node. The "ant" keeps track of the frequency of visiting/hitting nodes for an arbitrary sample[9] of hits.

20.13.1   Active ant options

While active, a number of ant options are presented in a top right prompt as follows,

> **probabilistic ant: quit-q speed-$</>$ pause-p**
> **rndnode-r redraw-d**
> **showhits/options-h showant(ON)-a rndant(OFF)-s:**

|	
options ...	*what they mean*
quit-q ...	quit the ant routine and return to the options in section 20.9.
speed-$</>$ ...	enter "$<$" to slow the ant down, or "$>$" to speed it up.
pause-p ...	to pause the ant — a top-right inset will appear, for example **node=7, new:** showing the current node hit, and the chance to enter a new start node — enter **return** to resume.
rndnode-r ...	restart the ant at a node selected at random — continue to keep track of node visits.
redraw-d ...	redraw the graph to delete the ant trail — continue to keep track of node visits.
showhits/options-h ...	pause the ant to show the frequency of ant-hits so far, and other options — see section 20.13.2 for more details.
showant(ON)-a ...	toggle the display of the ant and its trail. The current status (ON or OFF) is indicated. If the ant graphics are off, the statistics of hits are gathered much faster.
rndant(OFF)-s ...	toggle between a continuous ant (the default) and restarting at random nodes for each step. The current status, ON or OFF, is indicated — see below for more details.

With **rndant(ON)** a different kind of sample is produced; at each step the ant is re-started at a random node, but biased by node-weight. Thus the ant can hit all reachable nodes even in separate components. By contrast, the default **rndant(OFF)** starts at a random node and follows a Markov chain, a continuous path according to the output probabilities in the graph. Some graphs are fully interlinked, but in a graph consisting of distinct components the ant will be trapped inside[10] one of the components.

---

[9]The maximum sample of ant-hits is the maximum size of an unsigned long int -1 = 4294967295. If this is reached "Show ant-hits" (section 20.13.2) is activated automatically.

[10]Note that the eigenvectors of the jump table of a component (which may be the whole graph) should be the same as the list of hit frequencies of the component.

With **rndant(OFF)** the ant can become stuck at a node with just one output to itself, or a node with zero outputs. In the latter case the hit-promp, section 20.13.2 below, will activate autmatically, with the added message **- no targets, and stuck at node 3** — for example.

20.13.2   Show ant-hits
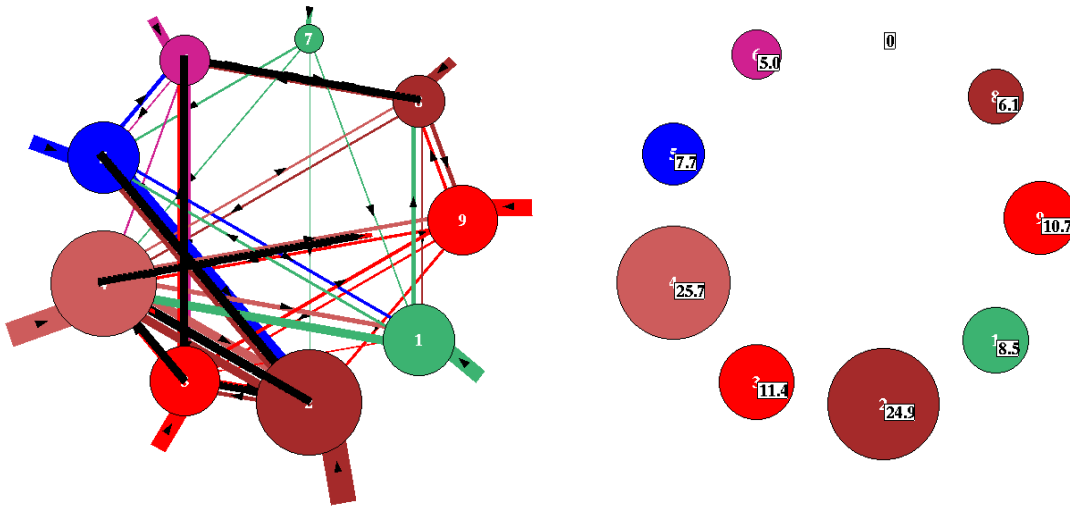
(a) The ant has a red body and leaves a black trail. Soon after launch the ant was paused while moving (in slow motion) from node 4 to 9. The default animation speed is very fast but can be controlled with "$</>$".

(b) The percentage of ant hits at each node after 5 million moves — with ant animation disabled this took a few seconds. Node 7 indicated by $\boxed{0}$ is unreachable from other nodes so has zero hits.

Figure 20.20: (a) A probabilistic ant moving along directed links between nodes. (b) Ant hits, or node visits (enter "**h**" followed "**n**"). Ant hits can be compared with basin (node) volume in (a) and the jump-table in section 20.2. This example is for the jump-graph of the RBN $v2k3$, $n=10$, defined in figure 20.11.

Enter "**h**" in section 20.13.1 while the probabilistic ant is active (or stuck) to show the number of hits so far and the percentage at each node. The nodes will be redraw and scaled according to hit frequency — edges, and nodes with zero hits are not shown. The following top-right options are presented, for example,

> **showing % hits on each basin** *(or* **each node** *for a network or ibaf graph)*
> **total hits so far=1453697 rndnode-r rndant(OFF)-s**
> **show % frequency-n noNodes-N quit-q cont-ret:**

options . . .	*what they mean*
total hits so far . . .	total hits when pausing "**h**" or autmatically if hitting a node with zero outputs with "**rndant(OFF)**".
rndnode-r . . .	restart the ant at a node selected at random — continue to keep track of node visits.

rndant(OFF)-s ... toggle between a continuous ant (the default) and restarting at a random (biased) node for each step. The current status, ON or OFF, is indicated as described in section 20.13.

show % frequency-n ... to add the frequencies of hits (as a percentage of total hits so far) to the scaled nodes (as in figure 20.20).

noNodes-N ... to show just the frequencies without showing the nodes, which might be necessary if large nodes obscure the data.

quit-q ... quit the ant routine and return to the initial-graph options (section 20.9).

cont-ret ... enter **return** to exit the ant-hit-options and continue accumulating ant-hits, returning to the active ant-prompts (section 20.13.1).

20.14   Redraw basins at jump-graph nodes
initial options, f-jump-graph only — see section 20.7.1

When drawing the basin of attraction field, the number of basins, their attractor periods, transient lengths, in-degree — so the sizes and shapes of basins — are hard to predict. A compact layout in the main window without overlaps can be somewhat difficult using the methods in chapter 25. This is solved by the ibaf-graph, but a previous method, to draw basins at the nodes of the f-jump-graph (with or without edges) retains some advantages. It allows geometric (circle/spiral/1d/2d/3d) or any other layout, which can be saved as a PostScript file if this option was set in section 24.2. The detailed basin layout can also be saved — the positions of each state in the uncompressed basin of attraction field — for loading back into the ibaf-graph (section 20.14.1).

The options for drawing basins at nodes with "**basins-i/I/s**" in the f-jump-graph initial reminder (section 20.7.3) are explained below.

prompt ... *what it means*

basins-**i** ... to redraw the basins centered at the jump-graph nodes positions, but without the jump-graph itself — suppressing nodes and/or edges, as in figure 20.21.

basins-**I** ... to keep the jump-graph, drawing basins within the nodes as in figures 20.22(a) and 20.23.

basins-**s** ... to rescale the basins to be redrawn in the jump-graph, the following top-right prompt is presented, for example,

> **change basin scale for graph (rad main window=38.9**
> **select attractor rad (min 0.24 def 19.0):** *(if the default was changed)*

Enter **return** to accept the default, or enter a new attractor radius which updates the default for the current jump-graph session. This can be estimated in relation to "**rad main window**" set in section 25.2 which is not affected (check with "**win-w**"), or the current jump-graph default.
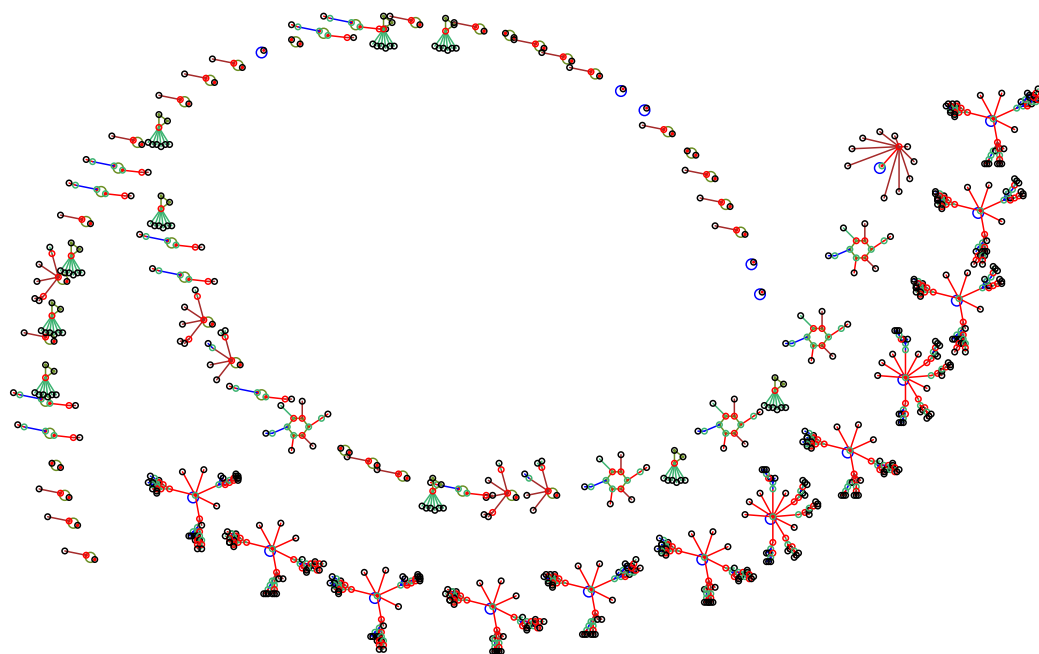
(a) all 73 basins — nodes and fragments were dragged starting from a circle layout.

Figure 20.21: Inserting basins in the jump-graph for a 1d CA $v2k3$, $n=10$, rule (dec)133, (a) uncompressed, (b) compressed. For a 1d CA, if edges were omitted in the f-jump-graph (section 20.7.1) the basin of attraction field can be compressed (section 26.2) to include just the prototype (non-equivalent) basins — otherwise all basins are shown. This provides an alternative layout method to the ibaf-graph, which does not allow compression.

(b) the 11 prototype basins, from a square layout.

20.14.1   Saving jump-graph basins to reload in ibaf-graph

For an uncompressed basin of attraction field, state-space coordinates (drawn in the f-jump-graph) can be saved, to reload into the ibaf-graph. Once basins are drawn, the top-right basin data window (section 27.2.1) will pause, with the following prompt in red,

> **PBC basin types=15 total basins=15** (*for v2k3 1d CA rcode 110, n=10*)
> **n=10 field=1024 g=442=0.42 0.033sec cont-ret, savelayout-s:**

Enter *savelayout*-**s** to record the coordinates of each state in state-space (a .grh file — section 20.16) — which can later be loaded back into the ibaf-graph of the same system, with its inital option *file*-**f** (section 20.6.2) to reproduce exactly the same layout (figure 20.22).

(a) circle layout jump-graph with basins drawn inside nodes.

(b) ibaf-graph with the jump-graph basin data loaded from (a) has exactly the same layout as (a).
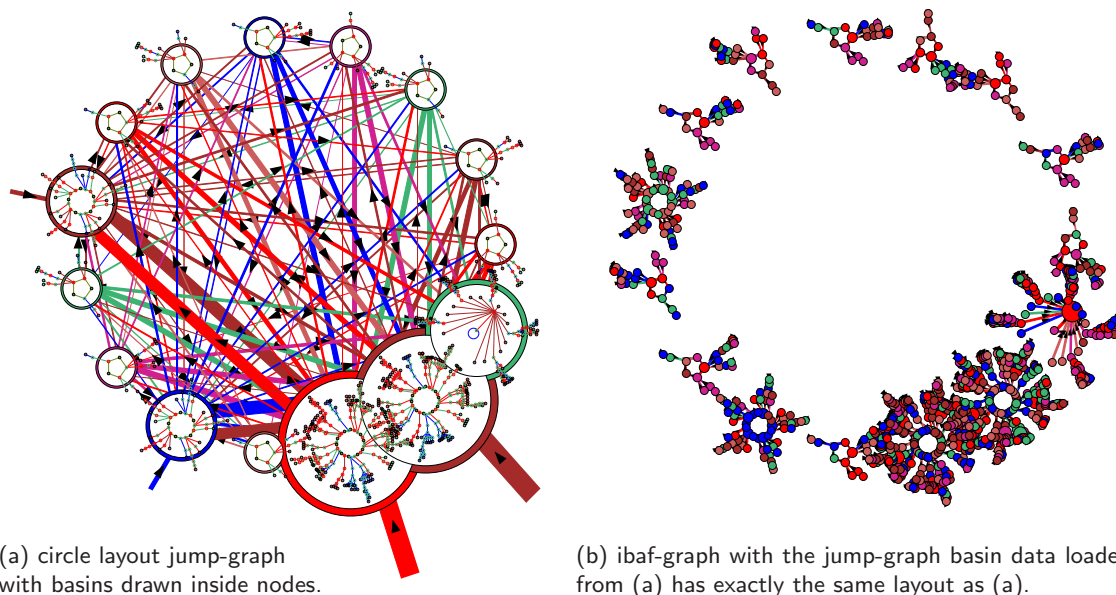
Figure 20.22: Reloading jump-graph basins coordinates (`.grh` file) into the ibaf-graph for an exact copy of the state-space layout, in this case ibaf basins in a circle. 1d $v2k3$ CA rcode 110, n=10

A file state-space not equal to ibaf-graph state-space will givae a top-right warning,

can't load: file nodes=1025, graph nodes=2049, cont-ret: (*for example*)

Whereas a matching state-space but for a different system will load, but produce scrambled links.

20.14.2 PostScript of jump-graph basins

Creating a vector PostScript file of attractor basins, whether drawn in the main windows or in the jump-graph, is activated in the "basin parameters" sequence of prompts, option "**PScript-P**" in section 24.1. To go directly to the PostScript prompt enter "**P**" at the first output parameter prompt, or arrive there by viewing the basin parameters in sequence.

The following top-right prompt is presented (see also section 24.2),

if saving to PostScript is currently active
save basin to postScipt (now p): greyscale-P color-p cancel-0: (*for example*)

if saving to PostScript is currently inactive
save basin to postScipt (now OFF): greyscale-P color-p: (*for example*)

Enter "**0**" to deactivate, "**P**" or "**p**" to activate — a filename prompt will be presented (section 35.3). If saving to PostScript is active, a new file will be created and overwritten to the selected filename each time an attractor basin is drawn (default filename `my_bPS.ps`). To conserve the file of the basins in the jump-graph, the default filename should be renamed with utilities[11] outside DDLab before a new attractor basin is generated.

---

[11]For example, rename in a terminal, look at the PostScript file in "GhostView".
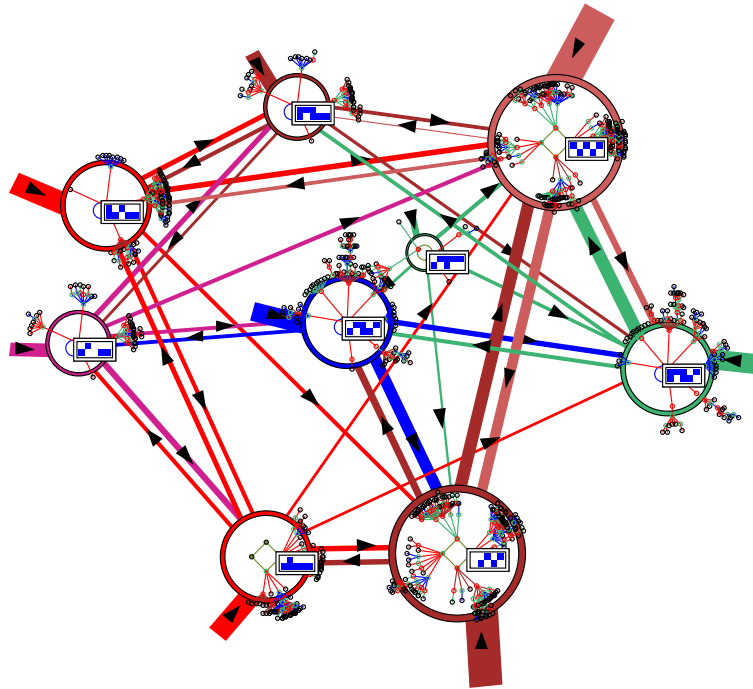
Figure 20.23: Basins of attraction redrawn at the jump-graph nodes, retaining the jump-graph itself. In this example one state in each attractor is also displayed. To create this figure as a vector PostScript file, the separate jump-graph and basin PostScript files were combined. This example is for the RBN $v2k3$, n=10, defined in figure 20.11.

The PostScript files of the attractor basin and the jump-graph (section 20.15) are two separate ASCII files, but can be combined (as in figure 20.23) by appending the basin file (without headers) at the end of the jump-graph file in an external editor. Alternatively, concatenate the files in the terminal — "`cat my_jgPS.ps my_bPS.ps > combined_file.ps`" — for example. .

20.15   PostScript of interactive-graphs
both initial and drag graphs

Enter "**P**" at either the initial-graph or drag-graph to save the current graph image as it appears — a vector PostScript file (chapter 36). The following top-right prompt is presented,

PostScript: greyscale-P color-p:

Enter "**P**" for greyscale or "**p**" for color. A filename prompt will be presented (section 35.3). The default filenames are `my_ngPS.ps`/`my_bgPS.ps`/`my_jgPS.ps` for network/ibaf/jump graphs. Once saving is complete the program reverts to the graph reminders.

The PostScript image can be viewed in GhostView — enter "`gv &`" in the terminal (chapter 36). For nodes as numbers or labels, there may be small differnces in text size between the screen and the PostScript images.

20.16   Graph layout file
initial-graph

It is sometimes useful to save a graph layout where time and effort have gone into a particular arrangement such as an unravelled network-graph, a rearranged ibaf-graph, or a jump-graph designed for redrawing basins at node positions (section 20.14).

Enter "*file*-**f**" at the initial-graph reminder (sections 20.5.3, 20.6.2, 20.7.3) to save the current graph layout, or load a layout where the number of nodes in the file and current graph must correspond. The following top-right prompt is presented,

   graph layout file: save-s load-l:

Enter "**s**" or "**l**" to save or load. A filename prompt will be presented (section 35.3). The default filename is my_grh.grh. When loading, if the number of nodes in the file and current graph do not correspond, the following top-right message is shown,

   can't load: file nodes=10, graph nodes=17: *(for example)*

If the file is successfully loaded, the current graph will snap into the new layout. The program reverts to the initial graph reminder.

20.17   Revise settings
initial-graph

Enter "*settings*-**S**" at the initial-graph reminder (section 20.9) to change the default settings for the rotation angle, the expand/contract factors for both nodes and links, and the arrow size on links. The following top-right prompts are presented in sequence,

   settings:rotation angle (start=15), now 15.00 degrees:
   expand/contract nodes: factor (1 to 2, start=1.2), now 1.20:
   expand/contract links: factor (1 to 2, start=1.1), now 1.10:
   change arrow size: (.03 to 1.0, start=0.07), now 0.07:

If required enter the new settings where "**start=x**" are reasonable initial values when starting interactive graphs. The allowed expand/contract factors for nodes and links range between 1.0 to 2.0. The current arrow size can be reset between 0.03 (tiny) and 1.0 (huge), which will take effect at the next contract/expand keyhit with "*arrows*-**</>**" — the expand/contract factor is constant at 10%, and arrows are toggled off/on with "*arrows*-**A**" (section 20.9).

20.18   Unreachable nodes
initial options

Enter "*Unreach*-**U**" in the initial-graph to show up unreachable or hard to reach nodes by disconnecting and/or isolating these node from the rest of the graph. A node is unreachable if it has no input links from other nodes in spite of possible output links — in the adjacency-matrix or jump-matrix (section 20.19), apart from self-links, there would be a blank column.

```
1:90
2:14
3:14
4:14
5:14
6:14
7:0
8:14
9:0
10:0
11:14
12:0
13:0
14:0
15:14
16:0
17:14
18:0
19:14
10:0
11:14
12:0
13:0
14:0
15:14
16:0
17:14
18:0
19:14
20:0
21:0
22:0
23:0
24:0
25:0
26:0
27:0
```
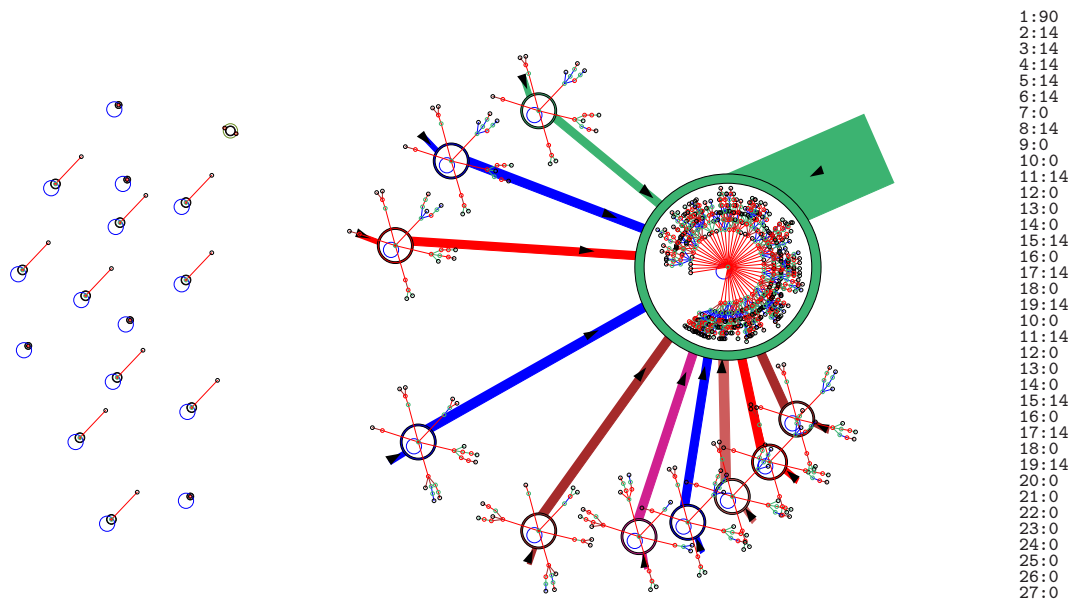
Figure 20.24: Disconnecting and isolating unreachable nodes. The disconnected nodes can be repositioned randomly on the far left side of the graph window, in this case with the default unreachability threshold of zero. This example is for a jump-graph of a 1d CA, $v2k3$ rcode 104, n=10. The basins of attraction were redrawn at the nodes (section 20.14). *Right*: The list of the number of jumps (inputs) to each basin (node) — shown in the terminal by entering "*xterm*-**x**".

An unreachability threshold of zero is the default for the jump-graph, and also for the ibaf-graph to isolate garden-of-Eden states, but for the network-graph the default is one because in DDLab each network node has at least one input. The unreachability threshold can be reset to a higher value, so that a node is considered "unreachable" if it has x or fewer input links. The following top-right prompts are presented in sequence,

> **unreachable nodes: xterm-x suppress-u restore-(ret):**
>
> *if "suppress-***u***" is entered, the following further prompts are presented*
> **isolate unreachable -i:**
> **change unreachable threshold (now 0):** *(for the network-graph (***now 1***))*

Enter "*xterm*-**x**" to list in the terminal the number of inputs to each node, to help select a threshold. Enter "*suppress*-**u**" to suppress all links to nodes whose input links are at or below the current threshold, or **return** to restore the links. If "*suppress*-**u**" was entered, at the next prompt, enter "*unreachable*-**i**" if you wish to isolate the unlinked nodes, repositioned randomly on the left side of the window. At the next prompt a new unreachability threshold can be set. In the resultant disconnected graph, invisible links are still functionally present so dragging fragments is not affected by the disconnect. The original graph can also be restored with "*net*-**#**".

20.19   The adjacency-matrix
initial options

Enter "**t**" or "**T**" in the initial options to toggle between an interactive-graph and its corresponding square adjacency-matrix (also termed a "jump-table" for the jump-graph) where rows give outputs and columns inputs, with the origin top-left. For the network-graph and jump-graph "**t**" shows the numbers of links and "**T**" the fractions of total links from each node (sections 20.19.1 and 20.19.2).

For the ibaf-graph (section 20.19.3) enter "**t**" to shows a pattern (of pixels or blocks) where rows show the single sucessor of each state (output), and columns show each state's pre-images (inputs) — there can be zero or more inputs. Enter "**T**" to show a list of exhaustive pairs in the terminal as in section 29.7.4.

The adjacency-matrix is presented in the graph window with various top-right options to amend/scan/save the presentation. Enter "**t**" or "**T**" again, or **return** to revert back to the interactive graph. The following sections give further details for each graph type.

20.19.1   adjacency-matrix — network-graph

```
        0   1   2   3   4   5   6   7   8   9    out   k
  0:  1   .       .   .       1   .   1   .   .    3    3
  1:  1   1   .   .   .   .   1   .   1   1    5    3
  2:  .   2   .   1   .   .   .   .   .   .    3    3
  3:  .   .   1   1   2   .   1   .   1   .    6    3
  4:  .   .   .   .   .   1   1   .   .   .    2    3
  5:  1   .   .   .   1   .   .   1   .   .    3    3
  6:  .   .   .   .   .   .   .   .   1   .    1    3
  7:  .   .   1   .   .   1   .   .   .   .    2    3
  8:  .   .   1   1   .   .   .   .   .   .    2    3
  9:  .   .   .   .   .   .   .   1   .   2    3    3
```
(a) rows show the numbers of outputs to other nodes.

```
        0    1    2    3    4    5    6    7    8    9   out  k
  0: 333   .        .    .        .  333   .  333   .    .    3   3
  1: 200  200   .   .        .        .  200   .  200  200    5   3
  2:  .   667        .  333   .        .   .        .    .    3   3
  3:  .    .   167  167  333   .  167   .  167   .    6   3
  4:  .    .        .    .        .  500  500    .        .    .    2   3
  5: 333   .        .    .  333   .        .  333   .    .    3   3
  6:  .    .        .    .        .   .        .   .    1    .    1   3
  7:  .    .   500    .        .  500   .        .   .    .    2   3
  8:  .    .   500  500   .        .   .        .   .    .    2   3
  9:  .    .        .    .        .   .        .  333   .  667    3   3
```

(b) rows show the fraction of outputs to other nodes — the decimal point is omitted and the precision depends on the column spacing which is adjustable. If all the outputs of a node are to just one other node, this is indicated by 1, as for node 6 with all its outputs to node 8.

Table 20.1:   The adjacency-matrix of the RBN $v2k3$, $n=10$, defined in figure 20.11. (a) The rows (0 to n-1) outputs, columns (0 to n-1) show inputs. Zero values ar shown as dots. (b) This can also shown as the fractions of each node's total outputs. The last two columns show the total outputs and inputs (k) for each node.

In an adjacency-matrix for the network-graph (table 20.1), rows (0 to n-1) show the numbers (or fractions) of output wires from each network element (node) to each node (columns 0 to n-1), so inputs can be read off columns. Two further columns headed "**out**" and "**k**" show the total outputs and inputs from/to each node.

20.19.2 adjacency-matrix — jump-graph

	1	2	3	4	5	6	7	8	9	P	J	Volume	Self
1:	5	.	.	3	1	.	.	1	.	1	10	110=10.74%	50.00%
2:	.	24	4	8	4	.	.	.	.	4	40	235=22.95%	60.00%
3:	1	5	24	2	.	4	.	.	4	4	40	106=10.35%	60.00%
4:	4	8	.	24	.	.	.	1	3	4	40	233=22.75%	60.00%
5:	1	3	.	.	5	1	.	.	.	1	10	108=10.55%	50.00%
6:	.	.	2	1	1	5	.	1	.	1	10	54=5.27%	50.00%
7:	3	3	.	3	4	.	7	.	.	2	20	18=1.76%	35.00%
8:	1	.	.	1	.	1	.	5	2	1	10	56=5.47%	50.00%
9:	.	1	1	1	.	.	.	1	6	1	10	104=10.16%	60.00%

(a) rows shows the number of jumps from each basin (rows) to each basin (columns).

	1	2	3	4	5	6	7	8	9	P	J	Volume	Self
1:	500	.	.	300	100	.	.	100	.	1	10	110=10.74%	50.00%
2:	.	600	100	200	100	.	.	.	.	4	40	235=22.95%	60.00%
3:	025	125	600	050	.	100	.	.	100	4	40	106=10.35%	60.00%
4:	100	200	.	600	.	.	.	025	075	4	40	233=22.75%	60.00%
5:	100	300	.	.	500	100	.	.	.	1	10	108=10.55%	50.00%
6:	.	.	200	100	100	500	.	100	.	1	10	54=5.27%	50.00%
7:	150	150	.	150	200	.	350	.	.	2	20	18=1.76%	35.00%
8:	100	.	.	100	.	100	.	500	200	1	10	56=5.47%	50.00%
9:	.	100	100	100	.	.	.	100	600	1	10	104=10.16%	60.00%

(b) rows show the fraction jumps to other nodes — the decimal point is omitted and the precision depends on the column spacing which is adjustable. If all the jumps from a node are to just one other node, this is indicated by 1,

Table 20.2: The jump-matrix for figure 20.23 and defined in figure 20.12. There are $n=9$ basins of attraction. (a) The rows (1 to 9) show the number of jumps from each basin to other basins (columns (1 to 9)). Zero values are show as dots. (b) Shows the fractions of each basins's total jumps.

In an adjacency-matrix (jump-matrix) for the jump-graph (table-20.2) with N basins in the basin of attraction field, rows (1 to N) show the numbers (or fractions) of jumps (outputs) from each basin to each basin (columns 1 to N), so incoming jumps (inputs) can be read off columns. Four further columns provide additional data as follows,

label ...	*what it means*
P ...	the period of the attractor.
J ...	the total number of possible 1-bit flips or jumps, which equals the network size multiplied by the attractor period, $n \times P$.
Volume ...	the total number of states in the basin of attraction, and its percentage of state-space, for example **110=10.74%**.
Self ...	the percentage of total jumps **J** that are self-jumps.

20.19.3 adjacency-matrix — ibaf-graph

For the ibaf-graph, enter "**T**" to show the list of exhaustive pairs in the terminal as in section 29.7.4. Enter "**t**" for the adjacency-matrix displayed as a pattern of $E=v^n$ unique successors (outputs) of each state in state-space within an $E \times E$ unnumbered matrix with the 0-0 origin top left — growing exponentially with system size n. Initially the matrix is confined within a fixed sized frame that fits comfortably within the matrix window, but where pixels might overlap, but this can be expanded to exclude overlap (stage +1) giving a non-overlapping pixels pattern. Further stages (+2, +3, etc) give square bars — 2×2 pixels, 3×3, etc. The non-overlapping patterns will escape the matrix frame, but there are options to jump to a new origin or move the pattern up/down and left/right, or to contract back.
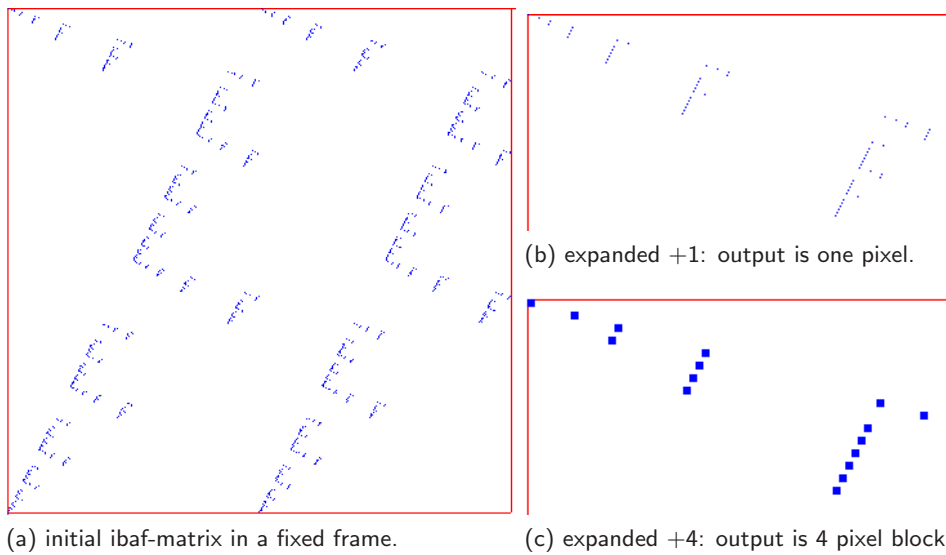
(a) initial ibaf-matrix in a fixed frame.

(b) expanded +1: output is one pixel.

(c) expanded +4: output is 4 pixel block.

Figure 20.25: The ibaf-matrix of the ibaf-graph for the $v2k3$ CA rcode 30 n=10 with $E=v^n$=1024 states in 6 basins (ibaf-graph figure 20.26). Compare the pixel-pattern with the return map for the same rule (figure 31.5). (a) The initial $E{\times}E$ matrix is confined within a fixed size frame. The matrix can be expanded, showing as much as will fit. (b) The expanded matrix, still with single pixels — top left area. (c) Expanded to 4-pixel blocks — top left area.
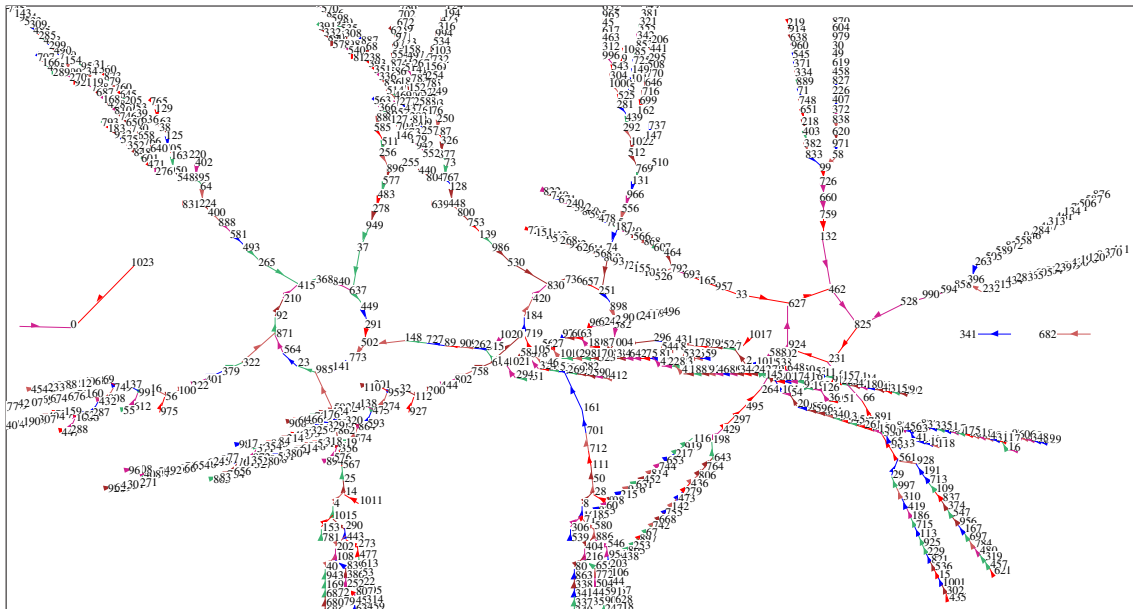

Figure 20.26: The ibaf-graph showing numbers, $v2k3$ rcode 30 n=10. Its ibaf-matrix is shown in figure 20.25.

In figure 20.25 (of the ibaf-graph in figure 20.26) unnumbered rows (0 to E-1) show a state's single sucessor (output) to unnumbered columns (0 to E-1). Pre-images (inputs) can be read off columns and their absence in a column denotes a garden-of-Eden state.

The resulting pattern has fractal characteristics — — a signature of the dynamical system, very similar if not identical to the "return map by value" (figure 31.5) constructed from space-time patterns running forward (section 31.2.2.2).

20.19.4   adjacency-matrix options

For all three graph types, when the adjacency-matrix is visible, the following top-right options allow the matrix to be scanned to see all parts of a large table where the data does not all fit inside the window, or to printed the data in the terminal,

> **adjacency-matrix: exit-ret xterm-x exp/cntr(2)-e/c**
> **jump(0-1023)-j down/up/right/left-d/u/r/l origin-o**

options ...	*what they mean*
exit-ret ...	exit the adjacency-matrix and return to the interactive graph and prompts in section 20.9.
xterm-x ...	print to the terminal — for the network and jump graphs: the adjacency-matrix, for the ibaf-graph: print the list of "state: successor" — which is the same as the "exhaustive pairs" list (section 29.7.4) but without state bits.
exp/cntr(0)-e/c ...	to expand/contract the matrix, where "**(0)**" indicates the initial (minimum) scale, and thereafter shows the current scale, "**(3)**" for example. For the network and jump graphs "**e**" expands the x coordinate (columns) by 1 pixel up to a maximum of 9 pixels, "**c**" contracts correspondingly. For the ibaf-graph and "**t**" selected in the initial options, both the x and y coordinates expand/contract by 1 pixel as well as the block representing the pattern of unique sucessors, minimum 1 pixel, maximum 30 pixels.
jump-j ...	jump to place a given row and column in the top-left position of the table. The following top-right prompt is presented,

<div align="center">

jump: row: column:

</div>

down-d ...	move down, increase the first row index.
up-u ...	move up, decrease the first row index.
right-r ...	move right, increase the first column index.
left-l ...	move left, decrease the first column index.
origin-o ...	redraw the matrix with the top-left origin starting at the first row and column.

20.20   Space-time patterns within the network-graph
initial options, f-jump-graph only (section 20.7.1)

Space-time patterns can be shown running within the network-graph layout (with links omitted) simultaneously to the normal presentation described in chapter 32. Enter "**g**" at the space-time pattern interrupt/pause prompt (section 32.16) to set up the network-graph while space-time patterns are running — section 32.19 gives further details.

    This allows enormous flexibility, as the network-graph has various default layouts: circle, spiral, 1d, 2d (square or hex), 3d, as well as allowing dragging nodes and components, and many other options for rearranging the graph (figures 32.42 — 32.43).

    Once the network-graph space-time patterns are running, and the normal space-time presentation is in 2d, diagonal scrolling can be set with on-the-fly key "**#**". To produce a scrolling image as in figures 20.27 or 4.10, suppress normal space-time patterns with on-the-fly key "**J**".
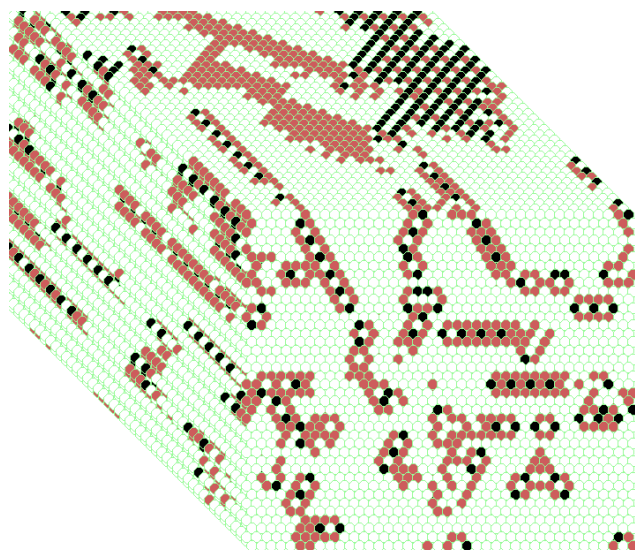

Figure 20.27: A 2d space-time pattern shown running within the network-graph layout, which has also been set to scroll diagonally. The present moment is the 2d pattern at the bottom-right.
2d CA 40×40, $v3k6$, kcode(hex) 0a502804958100.

20.21   network-graph from wring graphic — toggle
wring graphic initial options)

The interactive network-graph can be directly accessed from the wiring graphic (section 17.3) allowing toggling between the two. A new option **graph-g** is added to the (1d, 2d or 3d) wiring graphic reminder (section 17.4) to activate the network-graph — then the option **win-w** showing what was "below" the graph window provides a useful toggle between the two presentations as in figure 20.28. To resolve conflict, the wiring graphic option that was **all-g** is now changed to **all-a**.

To test, from a wiring graphic, enter '*graph*-**g**" for the following preliminary top-right prompt,

no links -N, show links -def:

Enter **return** to launch the network-graph with links, or "*no links*-**N**" without links. After an interval to compute, the network-graph and its iniitial reminder will appear, including the prompt "**win-w**" — enter "*win*-**w**" to toggle back (and forth) between the two without the need to recompute. Note that "**win-w**" is only active in the wiring-graphic initial-reminder. In the wiring-graphic there is a top-right inset "**restore graph -any key:**", so key **w** or any other key will work.
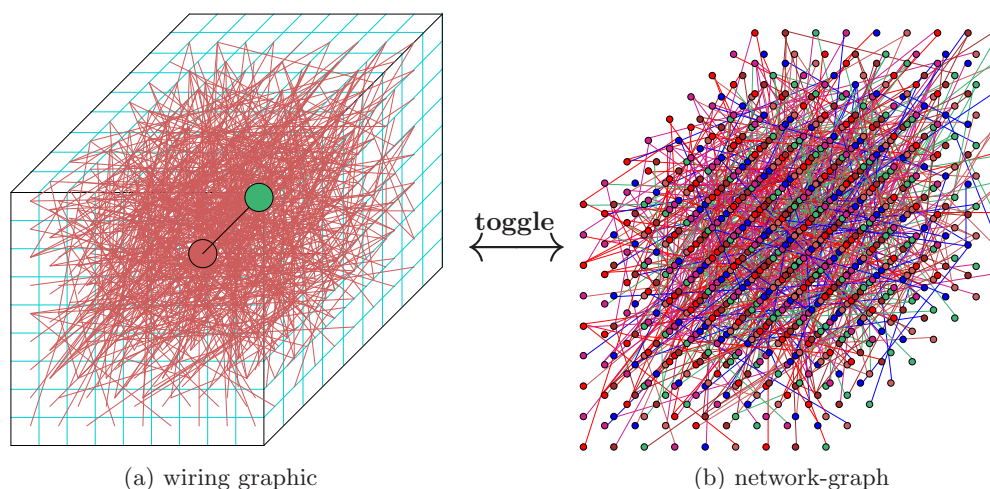

(a) wiring graphic   (b) network-graph

Figure 20.28: Toggling between the wiring graphic and network graph. This example is for a randomly wired 3d 9x9x9, $k=1$, network, and shows the wiring graphic (a) — its many options (section 17.4) now include "**graph-g**" which computes and initiates the corresponding interactive network-graph (b). Once visible, all network-graph interactive options (chapter 20) can be exercised, including "**win-w**" in the "initial reminder" to toogle back to see the wiring-graphic, where "**restore graph -any key:**" toggles back (and forth) to the network-graph without the need to recompute — the network-graph remains in control and active. To exit the network-graph and reactivate the wiring-graphic, enter **quit-q** from the initial reminder.