

Contents

- Visualizing the Local Search 2
 - Search Trajectory (ST) Visualization 3
 - Objective Value (OV) 7
 - Fitness Distance Correlation (FDC) 9
 - Event Bar (EB) 12
 - Algorithm-Specific (AS) Visualizations 13
 - Problem-Specific (PS) Visualizations 14

Visualizing the Local Search

Clearly, it is easier to understand large volumes of information if it can be presented graphically in an effective fashion, usually in form of 2-D graphs.¹ To demystify the bulk of information generated during local search runs, we have designed three forms of visualization in VIZ, namely generic local search visualization, algorithm-specific visualization and problem-specific visualization.

These visualizations exploit *human visual strengths* in understanding *spatial information*, in associating *colors, shapes* with information highlights, in observing *trends*, and in detecting *patterns, anomalies, outliers, discontinuities* and they are designed to follow the *Information Visualization Mantra* of: “overview first - zoom and filter - then details on demand...”

These visualizations are described in overview below and in details in the next subsections, where we will elaborate in details our design choices to create the visualizations that maximizes human visual perception strength, properly encode the visual information, and not taxing, e.g use a lot of short-term visual memory, eliminating any visual content that is not absolutely necessary to avoid visual clutter.²

Generic (or abstract) visualization is a powerful concept because it is independent of the underlying local search algorithm and combinatorial optimization problem. Generic visualization is possible in the context of local search for combinatorial optimization problems because every problem has a search space model [4, 7, 9, 3] and every local search algorithm works by mutating the current solution along this search space w.r.t generic properties such as objective values, distance, time (iterations).

VIZ has several generic visualizations: our main visualization — the **Search Trajectory (ST)** visualization, the standard **Objective Value (OV)** visualization augmented with several statistical information, the **Fitness Distance Correlation (FDC)** visualization which is essential for analyzing fitness landscape, and the **Event Bar (EB)** visualization, to highlight generic events that are occurring during the search. These visualizations are the fundamental analysis tool in VIZ, especially when the algorithm-specific and problem-specific visualizations are not available.

Algorithm-Specific (AS) visualizations refer to those that are specific to a particular local search algorithm. What is being visualized here is usually the dynamic parts of the local search algorithm. More precisely, we are interested to visualize the change in the values of the *dynamic* parameters over time³ which may explain why the local search behave like that at that point of time.

Some examples are: (i) observing the current temperature T in Simulated Annealing explains why a certain bad move is accepted or not accepted at that particular iteration; (ii) observing the tabu tenure over time in Tabu Search explains why the search trajectory quickly diverse away from a certain search region (during high tabu tenure phase) or concentrates/trapped in a search region (during low tabu tenure phase).

Problem-Specific (PS) visualization is the most intuitive⁴ form of visualization as it is directly related to the problem being solved.

There are several information that can be gained by observing problem specific visualization. We can get a glimpse of the problem structure which may be exploited for better performance, e.g. clustered versus distributed cities/customers in TSP/VRP. We can also verify the correctness of our problem-specific search strategy, e.g. we want to keep short edges in the TSP tour, did our local search algorithm really keep short edges. If the local search occasionally incorporates long edges, why it does that?

Problem-specific visualizations have their limitations. While some problems have natural visualizations, particularly those which can be cast in a spatial setting (e.g. TSP), it is not so clear how to do it in other cases (e.g. QAP/MKP).

There is also information overload since looking at the full solution has too much detail. Also it is harder to visualize the search trajectory since a problem specific visualization focuses too much

¹Of course, some information are still best explained using text. In VIZ, we combine our visualization with text-based output.

²What you see at the moment is a result of a lot of UI tweaks/testings... There were many visualization ideas that were dropped because they are not good enough...

³We don't actually need to visualize/animate static information as a static text based display is sufficient.

⁴In fact, a lot of approaches are concentrating on problem-specific visualizations: Some local search tools (e.g. COMET [12], a programming language approach for Constraint-Based Local Search) provide a built-in interface that allows the algorithm designer to implement problem-specific visualizations; Human-guided search [5], also rely a lot on problem-specific visualizations.

on the current solution in gory detail but does not show what is going on in the local search across a time interval, e.g. it might be difficult to see if the local search is trapped in a local minima simply by looking at several TSP tours or a tour animation.

Search Trajectory (ST) Visualization

Design of the Visualization

The concept of search trajectory visualization using anchor points has been described in details in the previous chapter. In this subsection, we describe how to implement that concept in VIZ.

After the points (anchor points and all the points in the local search runs) are laid out in the abstract 2-D space using the spring model layout algorithm described previously⁵, points that are drawn close to each other on the abstract 2-D space more or less have close distance in the real n-dimensional search space, and vice versa. Visualization in 2-D exploits human strength in understanding spatial information⁶, for example, points that are close to each other are readily perceived as being clustered/similar to each other (gestalt principle of proximity).

We don't consider extending this visualization into 3-D due to the following reasons:

1. Smaller dimension is easier to comprehend.
2. 3-D view can obscure things in the background (occluded by items in front).
3. Showing altitude by perspective drawing only adds clutter when compared to an overview from directly above with color coding, like what we did.
4. We have experimented with 3-D version of this visualization but it turns out to be an unsuccessful attempt as we don't have enough points to sufficiently render the 3-D space, thus it will be confusing for the human. Interpolating the meshes using Bezier interpolation is not that good as it is a lie to human eye.

Using only distance-based information does not tell all the story about local search behavior, as illustrated in Chapter 5.4.Z. We augment this visualization with objective value/fitness information. The points are drawn with different circular color label in the visualization screen to represent the objective value/fitness of the particular point w.r.t the best-known value (e.g. blue = 1% off — good; green = 1%..3% off — average; yellow = 3%..7% off — bad; without color = above 7% off — don't draw). This forms a 'contour map' and adjusting these range interactively can give us a lot of information about the fitness landscape/search behavior. Note that we purposely limit the number of category into 4 only (good, average, bad, don't draw) as this is near the limit (5) of human perceptual distinctiveness ability [2].

This Search Trajectory visualization by itself is an extension of FDC analysis (from comparison of fitness/distance w.r.t best found only, to comparison to multiple anchor points), however, augmented with the clearer FDC analysis, this is very useful for fitness landscape analysis (see Figure 1). For this, we are utilizing human strength in associating color with information as human visual perception classify objects that have similar visual attributes (color, shape, orientation) as belonging to the same group (gestalt principle of similarity).

Finally, to answer most of the questions about local search behavior, we need to visualize the search trajectory. As the local search moves locally (unless a strong diversification is performed), solutions found by local search that are close in time typically have strong connections. By connecting these solutions with lines, we have the trail of search trajectory. This is the temporal axis, the time dimension of our visualization.

This movement of the trail **line** of search trajectory is best displayed using an animation. The trail length is adjustable to provide different level of details, to differentiate the head from the tail, we fade the color, with the tail being dimmer (almost transparent) than the head (drawn in opaque color). We draw a circle or cross in the position of current solution to differentiate the uphill (cross) and downhill (circle) move. This gives an effective way of subtly showing uphill

⁵Due to the fact that we sort AP points based on decreasing fitness. Our layout algorithm will display the best found solution exactly in the middle of the visualization window (center of attention) and all other points are laid out step by step w.r.t the previously installed points. Therefore, the so called 'Big Valley' property, if any, will be easily identified by a bunch of good points in the middle of the screen.

⁶Of course, the loss of precision because of the mapping of high dimension to 2-D may render 'wrong conclusion'. However, so far such effect is minimal as the current users of VIZ reports successful understanding of what VIZ shows, as shown in Chapter ??.

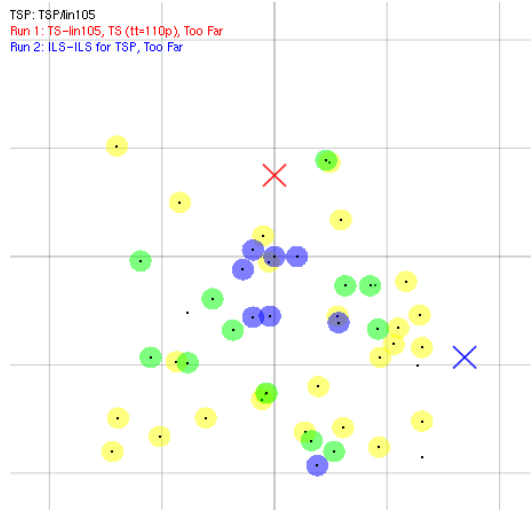


Figure 1: Search Trajectory Visualization — Fitness Landscape Overview, ‘X’ indicates the position of the initial solution.

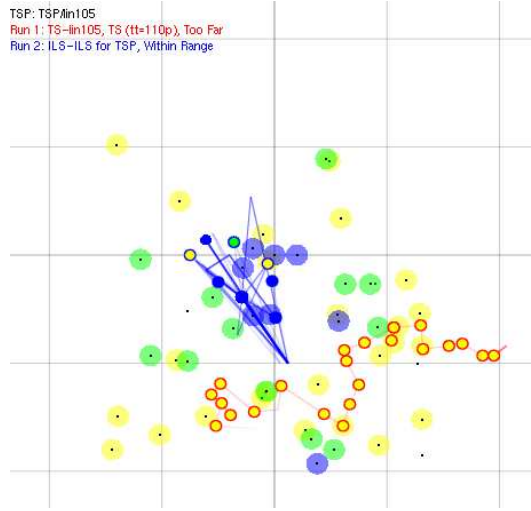


Figure 2: Search Trajectory Visualization. Two search trajectories Red (Strict-TS) and Blue (ILS) on TSP instance ‘lin105’ are visually compared.

and downhill moves without cluttering the animation with ‘rainbow color’. This utilizes human strength (gestalt principle of connection) to draw connection of the dynamic temporal sequence of the regions visited by the search at a particular point of time (see Figure 2).

Other than dynamic search trajectory visualization, we can also choose not to draw the connecting lines, but just highlight the points visited throughout the search trajectory in **area** mode. Using the color highlights in the contour map, we can see the quality of the region currently being searched in a glance (gestalt principle of similarity) (see Figure 3).

The **line** mode utilizes distance and time dimension whereas the **area** mode utilizes objective value dimension. We can combine both together in **detail** mode. However, we suggest that **detail** mode is used when the visualization is being paused/stopped as it is quite complicated to discern all the three dimensions at once.

As the search trajectory information when far from any point in the APSet is not meaningful, we decide not to draw that region of search trajectory. Therefore, when there is no line or circle (area) being drawn, it means that the search is currently exploring area far from the recorded APSet. The notion of ‘far’ is initially set to be the average distance from all points in search trajectory w.r.t to closest anchor point, but it is adjustable. This is to reduce misleading human visual perception system as search trajectory movement that are far from APSet are not that ‘meaningful’. We are planning to improve this section in the future.

Last but not least, in conjunction with the text-based information center, we allow the user

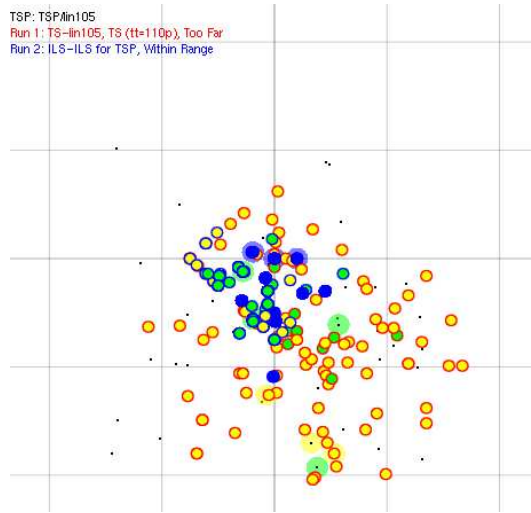


Figure 3: Search Trajectory Visualization — Search Coverage

to point and select (using a mouse) a particular point in the search trajectory visualization, and he/she can look at the details of that point (objective value, the solution itself, etc) in the text-based window.

Visualization Options

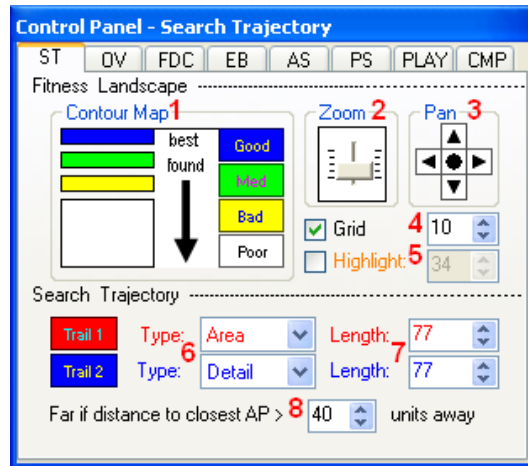


Figure 4: ST Visualization Options.

Interpretation of the Visualization

This primary visualization is very natural to explain the qualitative local search behavior⁷ w.r.t anchor points which may have been hidden without such visualization.

Almost all questions about the behavior of local search in Section ?? can be answered satisfactorily:

1. Does it behave like as what we intended?

Is the trajectory lines drawn behave like what we think it should behave? Whether the search is doing a correct/wrong thing at a particular iteration/period of time during the search can be argued via looking at the behavior of this trajectory line.

2. How good is the local search in intensification?

Look at the trajectory lines. Are they drawn inside one good local optima region ‘long enough’

⁷Our future aim: by looking at local search behavior only via Viz, the algorithm designer can describe the local search without knowing the algorithmic detail a-priori and a similar variant of that local search can be re-produced.

Items	Description
1	This is to determine the range of contour map. Drag the sliders to adjust. For contour map, we limit the differentiating color to be 4 levels (low — blue, medium — green, high — brown, and very high — no color/background color). This is to avoid rainbow-like visualization which is very confusing.
2	When we zoom in, we draw more details. When we zoom out, we draw an overview.
3	Panning ability will be more useful if zooming/level of details capability of Viz is improved... Currently, zooming and panning are to make ST visualization works for different monitor/screen resolutions only...
4	Grid lines with adjustable scale can be used to ‘gauge’ the distance information. The information gained by using grid lines is actually not 100% accurate as our layout algorithm yields approximate layout, handle with care!! The grid size can be adjusted... the green box denotes the diameter. The red box denotes the maximum possible drawing area.
5	This is to highlight the an AP in the ST visualization window. The information about this AP and its current problem specific visualization is immediately reflected in Text-Based Information Center and in Problem Specific visualization window, respectively. Use your mouse to hover a particular AP to see this feature in action.
6	Line is good to show temporal relationship (The solution quality is not embedded), Area is good to show search coverage+quality overview (This is usually used in non-playback mode, i.e. the playback is stopped...).
7	Detail is a semi-combination of both modes. This trail type option is differentiated for run1/run2. Trail length option is used to tradeoff between precision/detail (shorter trail length) versus overview (longer trail length).
8	This option is to adjust how far current solution should be from the set of anchor points so that it should not be drawn. The default value is [average distance + 7]. This is a new feature, more details will be written about it.

Table 1: ST Visualization Options

before it eventually moves away from such region? The duration/number of iterations when the lines stay longer in one region compared to when it quickly move from one region to another region is a rough measure of the intensification strength.

3. How good is the local search in diversification?

Look at the trajectory lines. When the local search is doing a diversification, is the line moves to a new unknown region, and then after that it converges/finds to a new/better local optima?

4. Is there any sign of cycling behavior?

Is the trajectory line comes back to a point (or a region) that was visited long time ago?

5. How does the local search algorithm make progress?

Explained via the movement of trajectory line. The geometric distance of the movement from s^t to s^{t+1} as seen on the screen (far/short) can be used to roughly gauge how radical/conservative the local search in modifying the solution per iteration. The sequence of visitation can tell something too. When the line is moving from good region to bad region, moving from infeasible region to a more feasible region, moving closer to constraint boundaries, or vice versa, we can infer some information.

6. Where in the search space does the search spend most of its time?

Where the line is drawn most of the time?

7. How far is the starting/initial/greedy solution w.r.t the global optima/best found solution?

The trajectory line will begin with a cross mark (indicator for the position of initial solution). How far is it from the center?

8. Does the search quickly find the global optima/best found solution region or does it wander around in other regions?

Starting from the initial solution (cross mark), how long does it take for the search to reach the center of the screen (best found solution). In our visualization, since we designed the best found solution to be in the center of the screen, the trajectory line will be able to get ‘close’ to the center of the screen but wanders around again to another place.

9. How wide is the local search coverage?

This is easily answerable using ‘area mode’ where we don’t draw the trajectory lines, but just highlight the points traversed by the search trajectory. By filtering bad points (by adjusting the contour map values), we can quickly identify where are the good points found by the search.

10. What is the effect of modifying a certain search parameter/component/strategy w.r.t the search behavior?

Compare the differences (if any) of the trajectory line of RunLog1 versus RunLog2 where RunLog2 is a slight modification of RunLog1.

11. How do two different algorithms compare?

Simply superimpose the two trajectories and compare the difference.

Known Limitations

1. The playback speed must be ‘slow enough’ for the user to grasp the information.
2. The search trajectory information is not meaningful during the period when the search is exploring search space far from any anchor points that we have (distance of current solution is $> x$ w.r.t all AP). This usually happens during ‘diversification’ phase or when exploring very rugged fitness landscape. Our current solution is simply don’t draw the search trajectory line during this period as it will be confusing otherwise.
3. We may mistakenly think this abstract 2-D space as real 2-D... We can have $d(A,B) = 10$, $d(B,C) = 10$, $d(A,C) = 10$ too... not true in metric space but it happens in N-dimensional space... think about this more carefully.

Objective Value (OV)

Design of the Visualization

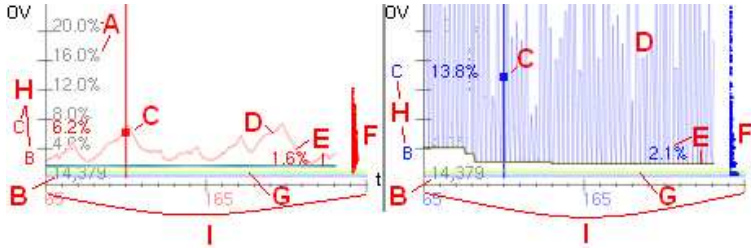


Figure 5: Objective Value over Time

Objective value (or fitness) is often the key⁸ attribute that drives a local search algorithm. Rather than simply plotting a conventional graph of the objective value over time/iterations (**C,D,H**), we enhance this with various statistical information to help understand the overall context of how the objective value is changing.

The extra information embedded in the objective value visualization are:

1. Max
In a maximizing problem, this is equal to the objective value of the best found solution, whereas in a minimizing problem, this value $Max = (100 + MaxDeltaFitness)\% * Min$. $MaxDeltaFitness$ is adjusted via option 3 for this visualization.
2. Min
In a minimizing problem, this is equal to the objective value of the best found solution, whereas in a maximizing problem, this value $Min = (100 - MaxDeltaFitness)\% * Max$. $MaxDeltaFitness$ is adjusted via option 3 for this visualization.
3. Frequency histogram (drawn vertically along y-axis in logarithmic scale)
This histogram is to highlight the average and distribution of objective values found by a local search run (**F**). Having such detailed histogram is better than just a box-plot.
4. A line to indicate best-found-so-far plus an indicator of percentage-off (the gap in percentage between current objective value w.r.t the best objective value found throughout the search run) (**E**).

⁸Perhaps, for some local searches, the only attribute

The values along Y-axis are computed w.r.t the best found objective value (**B**) and displayed either as percentage-off or absolute values (**A**). The best found objective value itself (**B**) will always be displayed as reference.

We purposely draw the current solution in the ‘middle’ of the objective value fluctuation graph (sparkline according to Tufte in [11]) so that the visualization is split into two parts: ‘immediate past’ and ‘immediate future’. We need both parts to give the flavor of the quality of the current solution within the context of its immediate past and immediate future.

We embed the same contour map colors used in Search Trajectory visualization in this Objective Value visualization too (**G**) so that the user can draw connections between the current search trajectory and its solution quality.

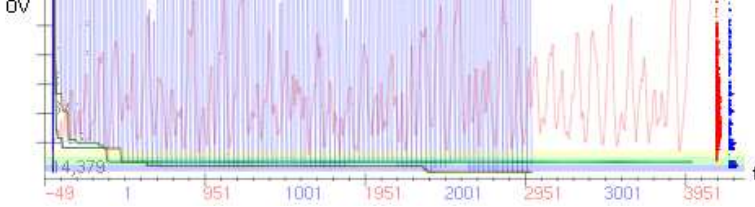


Figure 6: Objective Value visualization in overview mode.

User can use this visualization for conducting Run Time Distribution (RTD) analysis by scaling the x-axis (**I**) to fit the visualization window, omitting a lot of details (only display the ‘improvement graph’), superimposing two charts together, changing the playback mode by actual search time and then observing the resulting trends — something that human is good at (gestalt principle of continuity). In Figure 6, we can quickly see that even though the blue trajectory is very fluctuative, it manages to achieve better objective value than the red one.

Visualization Options

See Figure 7 and Table 2!

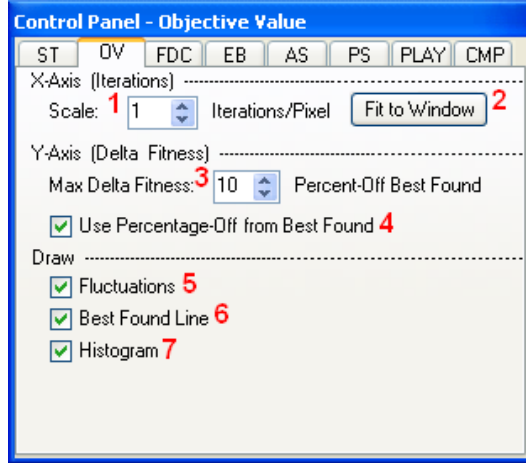


Figure 7: OV Visualization Options.

Interpretation of the Visualization

OV chart reveals information about local search behavior (section ??) from different angle:

1. Does it behave like as what we intended?

If we use Iterated Local Search which has pattern: perturbation - local search/improving - perturbation - ..., does our OV chart reflect this behavior? (like in Figure 5, right side).

2. How good is the local search in intensification?

Is the variance in the OV chart low but not cycling/repeated?

Items	Description
1,2	These options are used to scale the X-axis of the OV visualization. We can manually adjust the scale (1) or use the ‘Fit to Window’ button (2) to re-scale the whole entries of both trajectory 1 and trajectory 2 into the current width of OV visualization window.
3	This option enables the user to scale the Y-axis (delta fitness) especially around the values near best found objective value.
4	Sometimes, the absolute fitness is too big (occupy a lot of screen space). This option allows the user to toggle between displaying absolute value versus percentage-off from best found objective value to reduce visual clutter, e.g. ‘10,213,245’ vs ‘2.1%’. Displaying percentage-off is also more meaningful to the user for gauging how close/far to the current objective value w.r.t best found rather than using an absolute number...
5	This option allows the user to decide whether to draw the precise details of solution quality found throughout the search.
6	This option helps the user focus on the overall improvement steps found during the search.
7	This option allows the user to decide whether to display the histogram.

Table 2: OV Visualization Options

3. How good is the local search in diversification?

Can I see sudden jump in the OV chart that indicates a strong diversification?

4. Is there any sign of cycling behavior?

Is the pattern in OV chart repeated?

5. How does the local search algorithm make progress?

Using OV chart, we can only say the improvement progress (RTD analysis)

6. Where in the search space does the search spend most of its time?

Can’t be seen using OV visualization.

7. How far is the starting/initial/greedy solution w.r.t the global optima/best found solution?

See the OV differences between the first and the best OV.

8. Does the search quickly find the global optima/best found solution region or does it wander around in other regions?

Look at the OV fluctuations, when is it getting closer to best found OV?

9. How wide is the local search coverage?

Can’t be seen using OV visualization. However, we can use the histogram to explain the spread of solution quality found throughout the search.

10. What is the effect of modifying a certain search parameter/component/strategy w.r.t the search behavior?

Compare the differences (if any) of the OV chart of trajectory 1 versus trajectory 2 where trajectory 2 is a slight modification of trajectory 1. The important feature to compare is the difference in ‘best found line’ of both search trajectories.

11. How do two different algorithms compare?

Simply superimpose/juxtapose the two OV chart and compare the difference.

Known Limitations

1. The scale of x-axis is currently based on iteration. It is not sufficient as there is a potentially better scale: the actual search time.

Fitness Distance Correlation (FDC)

Design of the Visualization

FDC analysis is meant to give a rough measure of the problem difficulty w.r.t to the search space characteristics. In FDC analysis, we want to know whether there exists a correlation between the Fitness (F) and Distance (D) of the solutions w.r.t the *nearest* global optima (or best found solution). The FDC coefficient, r_{FD} (E) (See [4, 3]), is defined as:

$$r_{FD} = \frac{covariance_{FD}}{variance_F * variance_D} \quad (1)$$

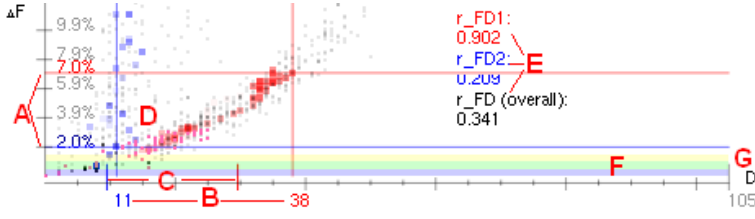


Figure 8: The FDC scatter plot of two search trajectories for TSP. TSP fitness landscape is known to have a ‘Big Valley’ property, thus its FDC coefficient is usually quite high.

$$cov_{FD} = \sum_{i=1}^n (F - \bar{F}) * (D - \bar{D}) \quad (2)$$

$$var_F = \sqrt{\sum_{i=1}^n (F - \bar{F})^2} \quad (3)$$

$$var_D = \sqrt{\sum_{i=1}^n (D - \bar{D})^2} \quad (4)$$

This r_{FD} information is gained by sampling a lot of local optima and then compute r_{FD} using these local optima. Compare the delta fitness and distance of each local optima to the *nearest*⁹ best found/global optima.

This FDC analysis is not perfect, but it is good analysis method to quickly portray search space properties. It is conjectured that for a minimization problem¹⁰, it is:

1. ‘straightforward’, when $r_{FD} \geq 0.15$, as the fitness increase as the local search is approaching global optima (\approx a linear line in FDC scatter plot).
2. ‘difficult’, when $-0.15 < r_{FD} < 0.15$, as there is very little correlation between fitness and distance w.r.t global optima. (\approx a line parallel to x-axis in FDC scatter plot).
3. ‘misleading’, when $r_{FD} < -0.15$, as the fitness decrease as the local search is approaching global optima. (\approx an inverse linear line in FDC scatter plot).

The FDC can be visualized using a scatter plot (**D**) by plotting the fitness difference along y-axis and distance along x-axis between the solutions¹¹ with the nearest best found. This scatter plot quickly shows whether there exist a correlation, be it a positive or negative one, and whether such correlation is strong enough (the points in the scatter plot are close to the trend line).

In VIZ, we ‘approximated’ the original FDC computation, which uses a lot of points, by compute r_{FD} with the ‘probably local optima’ points collected via RunLogs only. Thus, the quality¹² of our FDC computation depends on how many RunLogs are used and how diverse these RunLogs are.

Rather than just displaying a static FDC scatter plot, we can incorporate more information with an animation of the position of current solution w.r.t the nearest best found/global optima by plotting the F - D information over time (**A,B**). The position of this F - D is highlighted with a cross-hair. This is to gauge how good that particular run in traversing the search space, as shown by the movement of the points with inverted color against the backdrop of the overall FDC scatter plot in Figure 8. This FDC visualization is also augmented with the same contour map (**F**) as in Search Trajectory and Objective Value visualization.

While the y-axis (delta fitness) is adjustable based on the percentage-off w.r.t the best found objective value, the maximum size of the x-axis (**G**) is strictly equals to the to the diameter of

⁹As there exists a possibility that there are multiple best found/global optima, it is more appropriate to measure this FDC coefficient w.r.t the *nearest* one to avoid misleading the user by saying the search is ‘far’ from the target, thus low FDC coefficient, even though the search is actually already manage to arrive at ‘the other’ solution which has the *same* objective value/fitness with the best found.

¹⁰For maximizing, swap the definition for ‘straightforward’ with ‘misleading’ and vice versa.

¹¹The source of points are the local optima points in the RunLogs supplied to VIZ conversion wizard

¹²Note that both the standard FDC calculation and our simplified FDC calculation may be misleading if the actual global optima is/are not known, the best found solution(s) are used instead, and these best found solutions are, unfortunately, far from the true global optima.

the search space, that is, the size of the COP instance. This arrangement is useful to gauge the distribution of LO. TSP local optima are usually scattered around distance $\frac{1}{3}$ diameter (**C**) w.r.t the best found ('Big Valley') whereas most QAP local optima are scattered almost as far as the diameter, making QAP intrinsically a more difficult problem than TSP.

The result FDC analysis is usually uniform throughout various instances of a particular COP, e.g. almost all TSP instances have high FDC due to its 'Big Valley' property. However, for some COP, e.g. QAP, we may have different types, QAP type A (uniform) versus QAP type B (with flow-dominance).

Visualization Options

See Figure 9 and Table 3!!

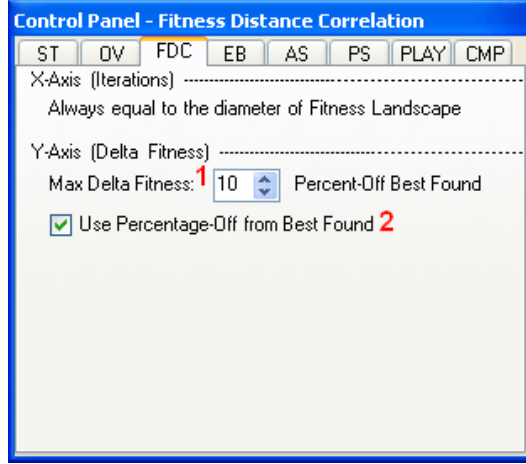


Figure 9: Fitness Distance Correlation Visualization Options.

Items	Description
1	Same as option 3 in Objective Value visualization.
2	Same as option 4 in Objective Value visualization.

Table 3: FDC Visualization Options

Interpretation of the Visualization

FDC chart mainly reveals basic information about fitness landscape which can give rough insights of local search behavior (section ??):

- Does it behave like as what we intended?**
The dynamic F-D plots can roughly say how local search is moving away or closer w.r.t best found solution.
- How good is the local search in intensification?**
Not clear in FDC visualization.
- How good is the local search in diversification?**
Not clear in FDC visualization.
- Is there any sign of cycling behavior?**
Not too clear in FDC visualization, but can be seen if quite obvious.
- How does the local search algorithm make progress?**
Not clear in FDC visualization.
- Where in the search space does the search spend most of its time?**
Not clear in FDC visualization.

7. **How far is the starting/initial/greedy solution w.r.t the global optima/best found solution?**
See the delta fitness and distance of the starting point w.r.t best found.
8. **Does the search quickly find the global optima/best found solution region or does it wander around in other regions?**
Not clear in FDC visualization.
9. **How wide is the local search coverage?**
Not clear in FDC visualization.
10. **What is the effect of modifying a certain search parameter/component/strategy w.r.t the search behavior?**
Compare the differences (if any) of the FDC chart of trajectory 1 versus trajectory 2 where trajectory 2 is a slight modification of trajectory 1. Usually in the spread of points in the scatter plot and the behavior of F - D positions.
11. **How do two different algorithms compare?**
Not clear in FDC visualization.

Event Bar (EB)

Design of the Visualization

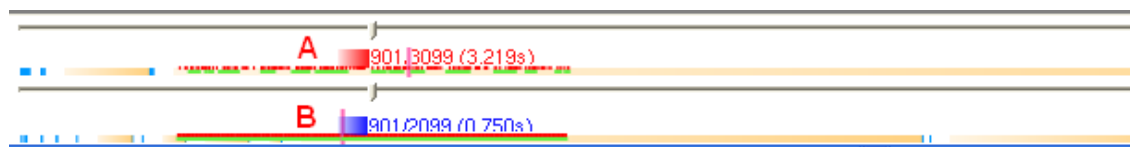


Figure 10: Event Bar

Like a video recorder, we employ index points/regions to assist the user in moving around in time during playback. The reason of using such highlights is because local search usually needs to run for a large number of iterations, e.g. thousands to millions. As such, the time bar¹³ alone may be either too fine or too coarse a granularity for moving around the search trajectory.

To highlight interesting portions of the trajectory, which may be missed/not obviously observed by human, VIZ automatically computes highlights (index points/regions) on *generic events*:

- ‘New Best Found’ — blue bars. Occurs if at iteration i , the search found a solution which has objective value better than iteration $[0..i - 1]$.
- ‘Series of Non Improving Moves’ — shades of orange. Occurs if after x iterations from iteration i , the search does not found a solution which has objective value better than iteration i . This event always occurs before ‘New Best Found’ event. Therefore they are drawn simultaneously.
- ‘Possibly Local Optima’ — red bars. Occurs when the pattern of objective value resembles a V shape or reverse-V shape, plus some plateaux variants. This indicator is only displayed locally $[-1,000 .. 1,000]$ around the current iteration.
- ‘Near set of Anchor Points’ — green bars. Occurs when at that iteration, the current solution is far (distance $> X$ units — X is adjustable from the ST visualization option) from any points in the AP set. This indicator is only displayed locally $[-1,000 .. 1,000]$ around the current iteration.
- ‘Has Tag Information’ — pink bars. Occurs when there is a tag information at that iteration. This indicator is only displayed locally $[-1,000 .. 1,000]$ around the current iteration.

¹³Time bar is our term for the two sliders that are used to jump anywhere throughout the playback of Search Trajectory 1 and 2.

These index points/regions are drawn in the event bar, which is attached to the time bar **(A)** & **(B)**. The time bar itself is augmented with an indicator for: [Iterations elapsed/Iterations left (Actual search time)] and a trail window to highlight which part is being shown on the screen at the moment. This visualization is best visualized side by side (juxtapose).

Visualization Options

See Figure 11 and Table 4.

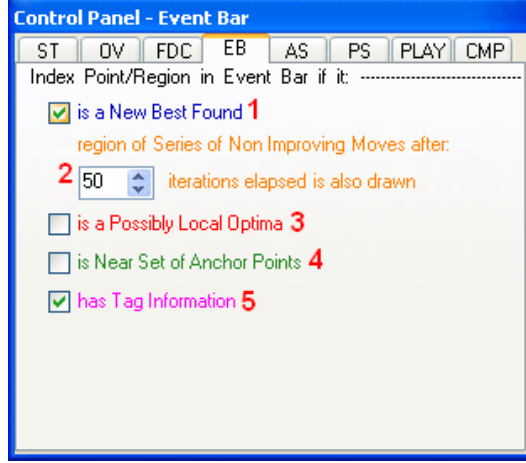


Figure 11: Event Bar Visualization Options.

Items	Description
1	A tag of small blue line is drawn in Event Bar at the position (iteration) where the overall best found is improved. This highlight is important to see whether the search is only improving at the early stages and never improves again (often encountered) or whether the search has potential to find better solution the longer the search goes on. This is again a form of RTD analysis.
2	Shades of orange are drawn in the Event Bar after X (adjustable by the numericUpDown control (2)) number of non improving iterations since last new best found occurred. This highlight is important to highlight ineffective period where the search is struggling to find new promising region.
3	A tag of small red line is drawn in Event Bar at the position (iteration) where the objective value pattern is as follows: down-up, or down-plateau, or plateau-plateau (for minimizing problem), and up-down, or up-plateau, or plateau-plateau (for maximizing problem).
4	A tag of small green line is drawn in Event Bar at the position where current solution is near the set of AP. The notion of 'far' is adjustable in ST visualization options.
5	A tag of long pink line will be drawn if there is a tag information for a particular iteration.

Table 4: EB Visualization Options

Interpretation of the Visualization

The sole purpose of this Event Bar visualization is to make it easier for the user to identify region of interest (as listed in the section about the option for this visualization) and to skip possibly boring parts of the animation. This visualization must always be used in conjunction with other visualizations (see Figure 10).

Algorithm-Specific (AS) Visualizations

Design & Interpretation of the Visualization

As its name implies, algorithm specific visualization depends a lot on the metaheuristic/local search being used. It will be near impossible to support *all*, therefore in the current version of VIZ, we limit ourselves to support visualization of the possibly dynamic part(s) of 3 local searches: the tabu tenure over time for Tabu Search (see Figure 12), the perturbation strength and the acceptance/rejection rate for Iterated Local Search (see Figure 13), and the temperature/acceptance probability for Simulated Annealing (see Figure 14).

To be meaningful, these algorithm specific visualization must be related with the information from other visualization(s).

Tabu Search

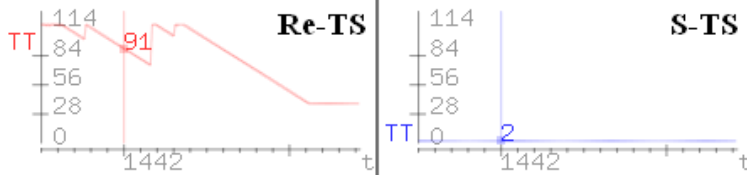


Figure 12: Tabu Tenure over Time for two types of TS, the details will be discussed in Chapter ???. The ‘TT’ along the y-axis refers to ‘tabu tenure’ at that iteration.

To use this visualization, record the Tabu Tenure values in the RunLog according to the 2006a format.

Iterated Local Search

Figure 13: Algorithm-Specific Visualization: Perturbation Strength and Acceptance/Rejection rate (ILS) — Figure N/A yet...

This visualization plots the perturbation strength at each iteration (usually constant pattern: 0,0,jump,0,0,jump...), and then tell whether after such jump, the new local optima are accepted or not afterwards. A summary of accept/reject rate is shown.

To use this visualization, record the Perturbation Strength and boolean of Accept/Reject values in the RunLog according to the 2006a format.

Simulated Annealing

Figure 14: Algorithm-Specific Visualization: Temperature and Current Random Toss (SA) — Figure N/A yet...

This visualization plots the Temperature over time, hence the acceptance probability (using Boltzmann function over time), and then compare the random toss value of that iteration w.r.t the acceptance probability. A summary of accept/reject rate is shown.

To use this visualization, record the Temperature and Toss Value values in the RunLog according to the 2006a format.

Visualization Options

See Figure 15 and Table 5.

Problem-Specific (PS) Visualizations

Design & Interpretation of the Visualization

Interpreting problem specific visualization should be straightforward. However, they will be more meaningful if combined with the information from other visualization(s).

However, with so many combinatorial optimization problems and their variants out there, it is also near impossible to provide support to *all* COPs. In the current version of VIZ, we support TSP and QAP visualization.

TSP

TSP solutions/tours (see Figure 16) are quite straightforward to visualize. The early work for visualizing TSP tours, especially to assist man-machine interactive optimization, was already begun in 1960s [8, 6]. More recent work is the Human Guided Search approach, e.g. [5].

Visualizing TSP tours like this enables human to quickly spot crossings, a ‘bad feature’ in a TSP tour. Computer algorithm needs $O(n)$ to draw this tour but it itself will require $O(n^2)$

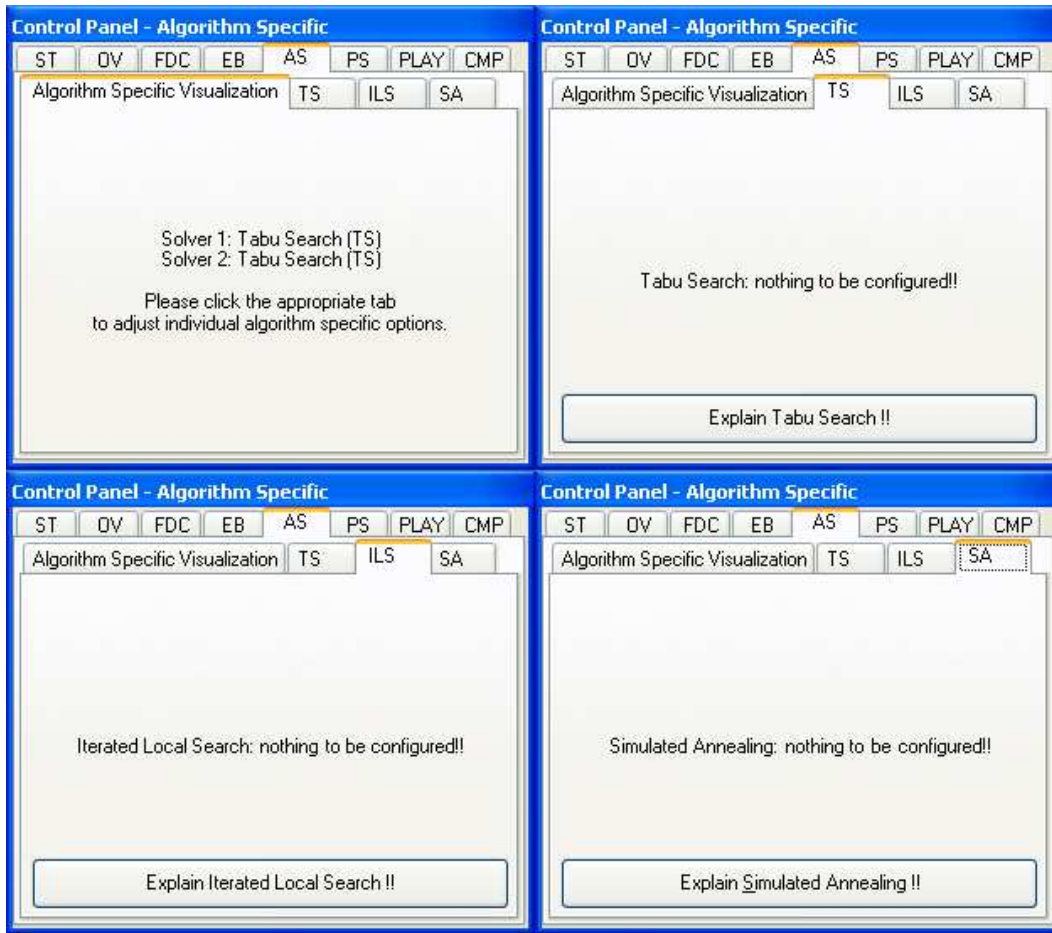


Figure 15: Algorithm Specific Visualization Options.

Items	Description
N/A	N/A

Table 5: AS Visualization Options

all-pairs line intersection tests (e.g. CCW test in Computational Geometry) to verify whether two lines in the TSP tour are crossing each other.

To use this visualization, please specify the path for the original TSPLIB instances (*.tsp) [10] in the ProblemSpecificFile field of your RunLog.

QAP

QAP solutions/permutations are not so easy to visualize. For this, we again employ the spring model layout algorithm to map the distance information into 2-D space. Currently, the PS visualization for QAP is the visualization of flow and distance matrix (see Figure 17).

To use this visualization, please specify the path for the original QAPLIB instances (*.dat) [1] in the ProblemSpecificFile field of your RunLog.

Visualization Options

See Figure 18 and Table 6.

Items	Description
1	Ability to juxtapose the solution with the solution of the highlighted AP.
TSP.1	Draw vertex ID
QAP.1	Decide whether to draw flow/distance matrix or the bipartite facility-location assignment

Table 6: PS Visualization Options

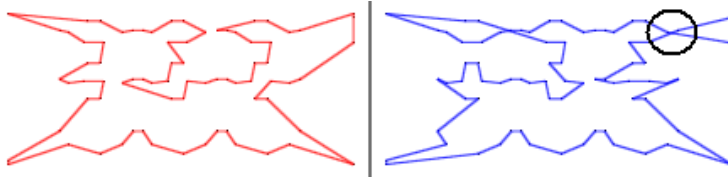


Figure 16: TSP tour visualization of instance pr76 from TSPLIB, left: optimal tour, right: non-optimal tour as there is an obvious crossing in the tour (circled).

Figure 17: Problem-Specific Visualization: QAP bipartite facility-location assignment.

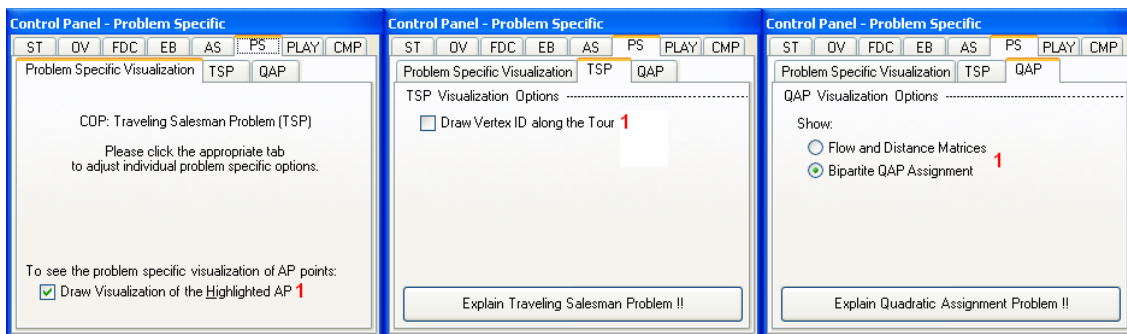


Figure 18: Problem Specific Visualization Options.

Bibliography

- [1] R.E. Burkard, S.E. Karisch, and F. Rendl. QAPLIB - A Quadratic Assignment Problem Library. *European Journal of Operational Research*, 55:115–119, 1991.
- [2] S. Few. *Information Dashboard Design*. O'Reilly, 2006.
- [3] H.H. Hoos and T. Stuetzle. *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann, 2005.
- [4] T. Jones. *Evolutionary Algorithms, Fitness Landscapes and Search*. PhD thesis, The University of New Mexico, Albuquerque, New Mexico, 1995.
- [5] G.W. Klau, N. Lesh, J. Marks, and M. Mitzenmacher. Human-Guided Tabu Search. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pages 41–47, 2002.
- [6] P. Krolak, W. Felts, and G. Marble. A Man-Machine Approach Toward Solving The Traveling Salesman Problem. *Communications of the ACM*, 14(5):327–334, 1971.
- [7] P. Merz. *Memetic Algorithms for Combinatorial Optimization Problems: Fitness Landscapes and Effective Search Strategies*. PhD thesis, University of Siegen, Germany, 2000.
- [8] D. Michie, J.G. Fleming, and J.V. Oldfield. *A Comparison of Heuristic, Interactive, and Unaided Methods of Solving a Shortest-route Problem*, pages 245–256. Michie, D. (eds). Machine Intelligence series 3. Edinburgh University Press, 1968.
- [9] C.M. Reidys and P.F. Stadler. Combinatorial Landscapes. *SIAM Review*, 44(1):3–54, 2002.
- [10] G. Reinelt. TSPLIB - A Traveling Salesman Problem Library. *ORSA Journal on Computing*, 3:376–384, 1991.
- [11] E. Tufte. *Beautiful Evidence*. Graphic Press, 2006.
- [12] P. van Hentenryck and L. Michel. *Constraint-Based Local Search*. MIT Press, Cambridge, London, 2005.