The Travelling Salesman Problem

Solved in Visual Prolog 7.5 language

by Ferenc Nagy, Budapest, Hungary.

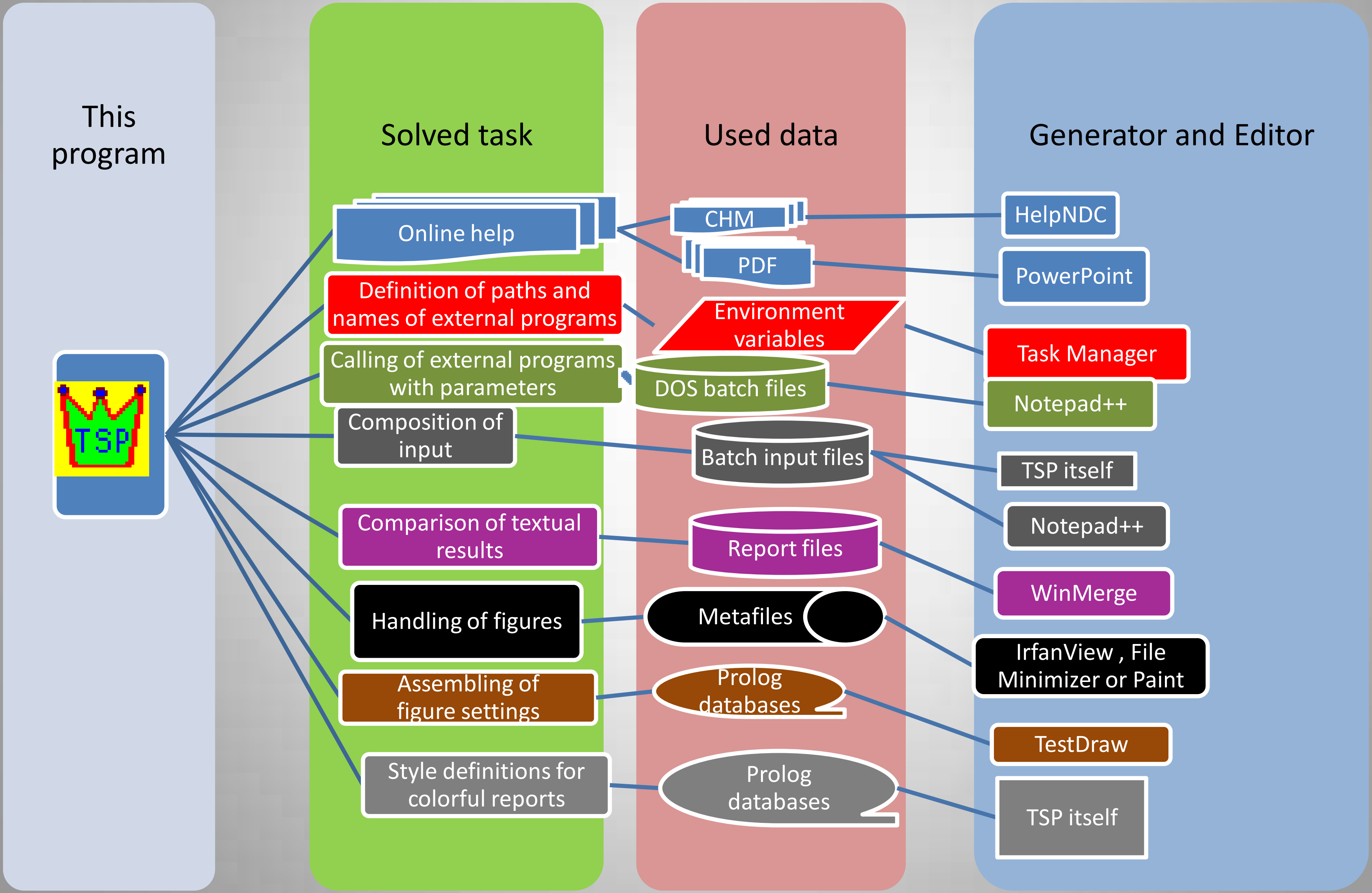
Date of last revision: October 11, 2015.

Chapter 1

Object Model

Next Chapter

Internal and External Resources of the Program



This program

Solved task

Used data

Generator and Editor

Online help – CHM – HelpNDC

Online help – PDF – PowerPoint

Definition of paths and names of external programs – Environment variables – Task Manager

Calling of external programs with parameters – DOS batch files – Notepad++

Composition of input – Batch input files – TSP itself

Composition of input – Batch input files – Notepad++

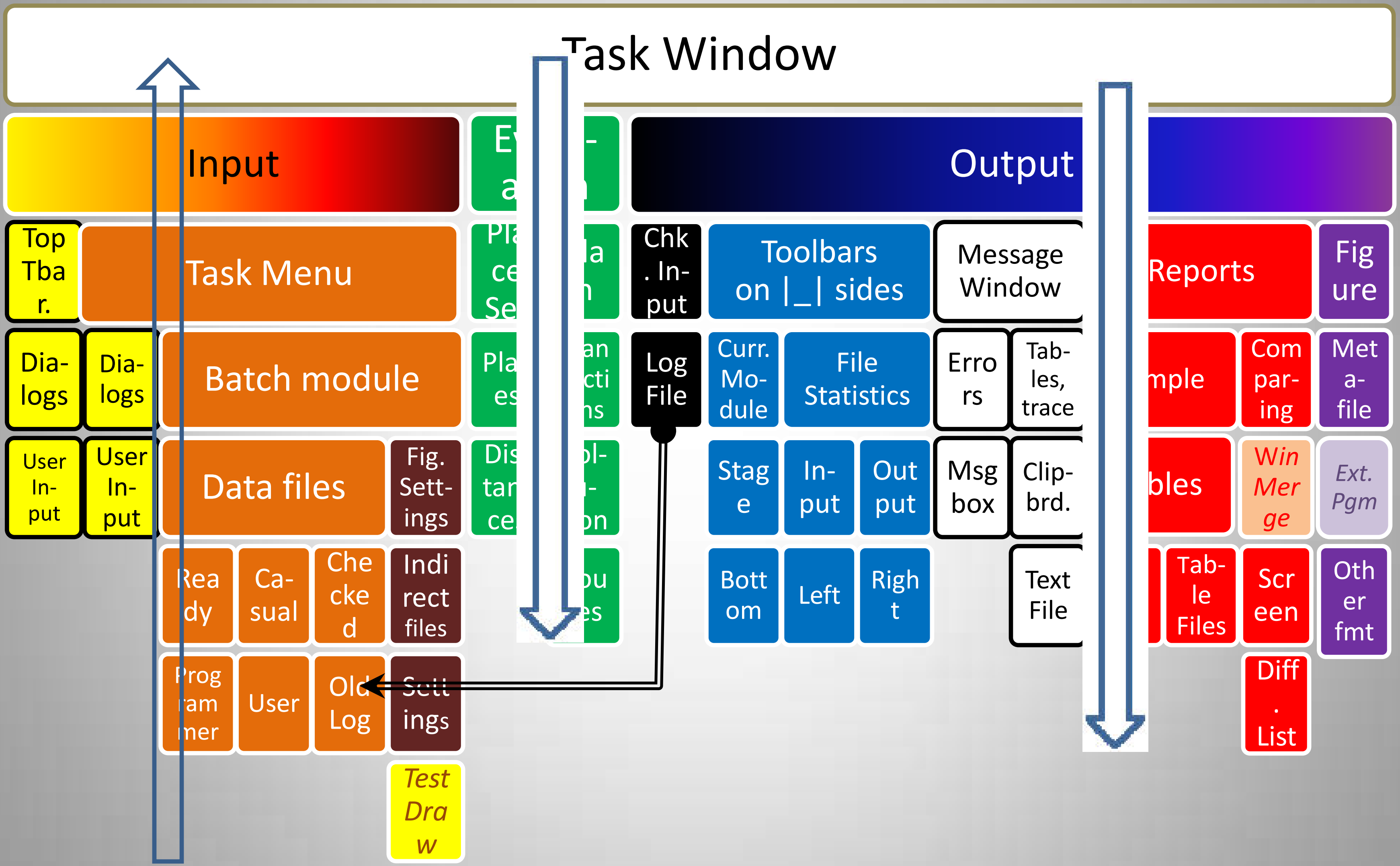
Comparison of textual results – Report files – WinMerge

Handling of figures – Metafiles – IrfanView , File Minimizer or Paint

Assembling of figure settings – Prolog databases – TestDraw

Style definitions for colorful reports – Prolog databases – TSP itself

Data Flow



Task Window

Input

Ev -
a n

Output

Top Tbar.

Task Menu

Pla ce Se
la n

Chk. Input

Toolbars on |_| sides

Message Window

Reports

Figure

Dialogs

Dialogs

Batch module

Pla es
an cti ns

Log File

Curr. Module

File Statistics

Errors

Tables, trace

mple

Comparing

Metafile

User Input

User Input

Data files

Fig. Settings

Dis tar ce
ol- - on

Stage

Input

Output

Msg box

Clipbrd.

bles

Win Merge

Ext. Pgm

Ready

Casual

Checked

Indirect files

pu es

Bottom

Left

Right

Text File

Table Files

Screen

Other fmt

Programmer

User

Old Log

Settings

Diff. List

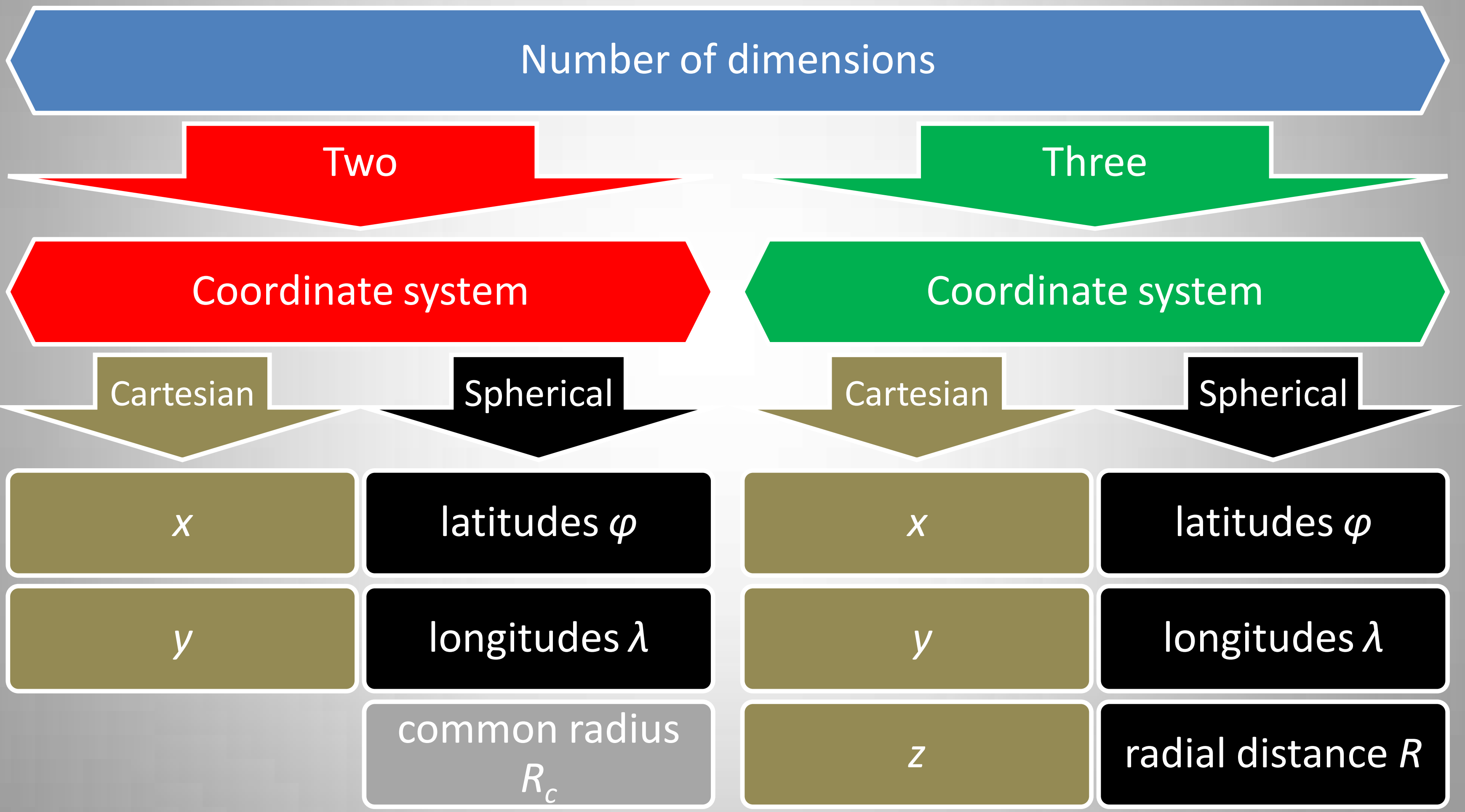
Test Draw

The Most Important Inherited Declarations

from FLAGS and COLLECT_SETTINGS Objects

	Domains	*Explanations*	*Entities*	*Name*	*Domain*	*Contents*
FLAGS	Identifier	String identifying the instance	Class properties	active_member	Identifier	Identifier of the active member of the collection of instances
	Protection	Allowed operations on instance	Instance properties	protection_flag	protection	Protection flag of the instance
	Variable	Editable.		validation_flag	validation	Validation flag of the instance
	Evaluated	Fixed		label	string	Long description of the instance
	Referenced	Fixed and must not be deleted	Instance predicates	formatLabel	string	Format labels for output
	Validation	State of variable		interpretLabel	-	Interpretation of batch LABEL commands
	Un-checked	Not yet read and checked	Constants	f_symbol_and_integer_unit	string	Same format for lists of fields having identical sequence of type
	Valid	Read and valid		f_longstring_ureal_unit		
	Invalid	Read and discarded		f_symbol_and_real_unit		
COL	Purpose	Symbols defining the usage of the settings	Constants	general_masks	string_list	File selection masks
				Indirect_masks		

Axes of the Coordinate Systems



Number of dimensions

Two

Coordinate system

Cartesian

x

y

Spherical

latitudes φ

longitudes λ

common radius R_c

Three

Coordinate system

Cartesian

x

y

z

Spherical

latitudes φ

longitudes λ

radial distance R

Place

I. Creation data

1. An identifier: NAME
2. The input coordinate system is common for all places of the set
3. , 4. [, 5.] Two or three coordinates depending on the coordinate system

II. Inherited properties

1. Status FLAGS

III. Methods for

1. Creation and modification from batch input files and interactive dialogs
2. Conversion between coordinate systems
3. Distance calculation
4. Classification of points supporting the projection of the data

Place Set

The place sets may be created

i. either in empty state

ii. or filled with places copied from another place set. These places are converted to the coordinate system of the target set.

I. Own properties

1. An **identifier**: TITLE
2. A **property:** the coordinate system of all added and replaced places.
3. A **collection** of PLACE OBJECTS having *p* members
4. Validating base on relative distance of the places

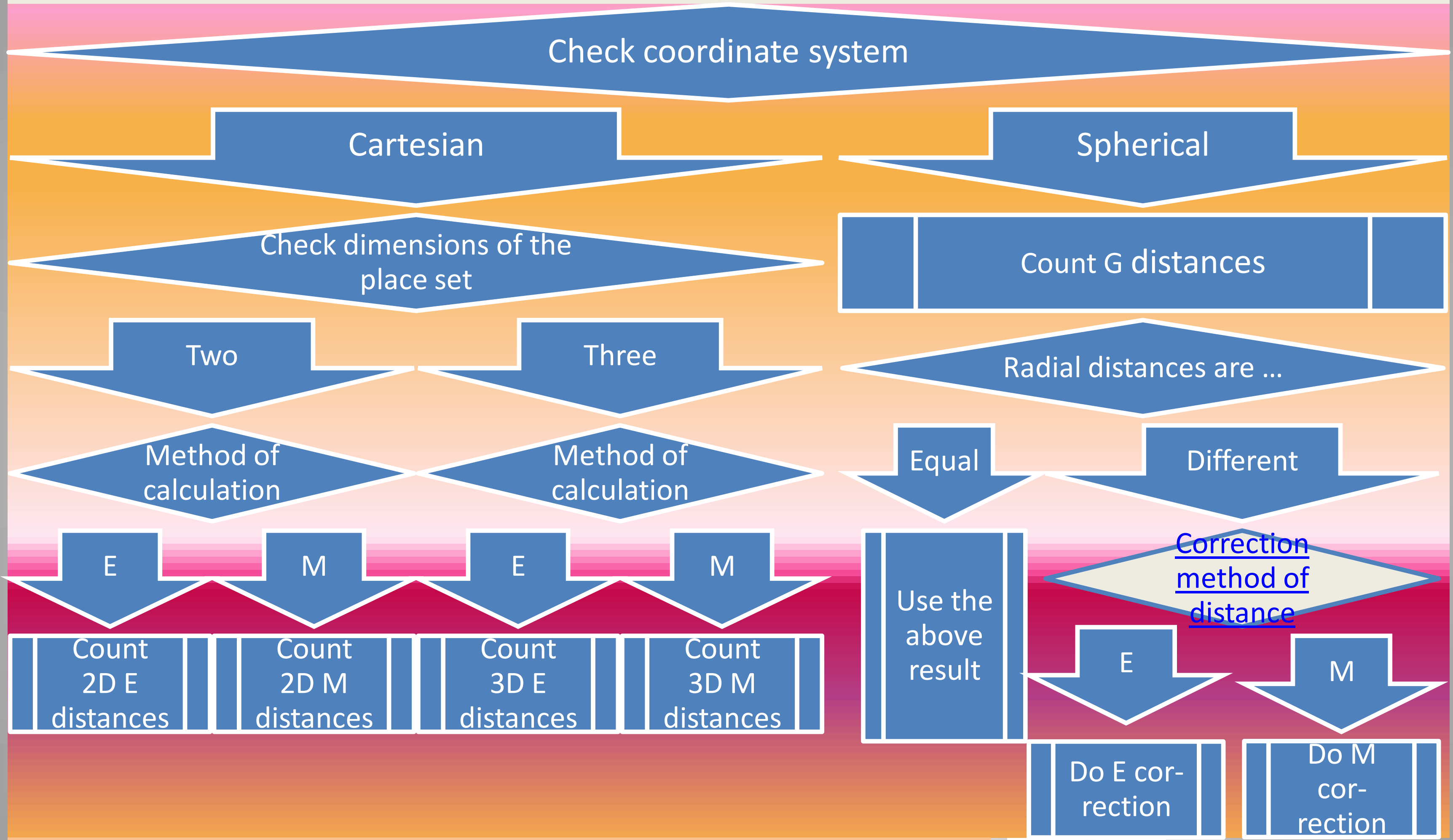
II. Inherited properties

1. Status FLAGS

III. They have just like as other objects

1. Procedures supporting their creation and modification
   a) from batch input files
   b) and interactive dialogs

Selection of Distance Functions



Check coordinate system

Cartesian

Check dimensions of the place set

Two

Method of calculation

E

Count 2D E distances

M

Count 2D M distances

Three

Method of calculation

E

Count 3D E distances

M

Count 3D M distances

Spherical

Count G distances

Radial distances are …

Equal

Use the above result

Different

Correction method of distance

E

Do E cor-rection

M

Do M cor-rection

See formulae at Algorithms

E = Eucledean M = Manhattan G = Great-circle

Plans and Transactions

I. The plans have
   1. An **identifier**: VERSION
   2. The kind of search:
      a) „full”,
      b) „greedy”§
      c) „undo”.
   3. A **collection** made of tuples of
      i. transactions defined for this kind and
      ii. allowed number of their repeat counts given as
         i. plain integers or
         ii. simple formulae if necessary¤.

II. The PLAN object inherit methods and properties
   – From the FLAGS object.

III. They have just like as other objects
   – Procedures supporting their creation and modification
      a) from **batch** input files
      b) and **interactive** dialogs.

IV. The plans may be created
   a) either in ***empty*** *state*
   b) or ***filled*** *with properties and the collection of* ***transactions from another plan.***

§: The fuzzy search uses the same transactions as the greedy one, only the method of the comparison of the gained distances is different from the greedy search. The greedy and fuzzy search is distinguished in the SOLUTION object.

¤: Some transactions can be executed only once in a plan, so they need not repeat counts.

General Properties of the SOLUTION Object

I. **Own properties and methods**

1. An **identifier**: VERSION
2. This object is based on a given PLACE SET
3. and a PLAN of transactions.
4. **Its main method** executes the plan leading to
    - a) a closed route around the members of the place set or
    - b) a set of opened routes.

II. **Inherited properties and methods:** from FLAGS object.

III. **They have just like as other objects**

- Procedures supporting their creation and modification
    - a) from **batch** input files
    - b) or **interactive** dialogs.

Special Properties of the SOLUTION

Object

I. **Initial state:**

   a) Set of unconnected places.

      i. the method of distance calculation.

II. **Restriction about the examined places**

   a) The members of the whole place set are examined.

   b) Those places are examined whose distances from a central place is less than or equal to a given threshold.

   c) Those places are examined whose distances from a central place is greater than a given threshold.

III. **Selection of the best solution**

   a) Greedy and deterministic: The smallest added distance and the alphabetical order of the places determines the added edges.

   b) Fuzzy and probabilistic: If more than one added edges result the same increase of the total length then the program randomly selects from them.

Basic Properties of the SOLUTION Objects

I. **Own properties and methods**

1. An **identifier**: VERSION
2. This object is based on a given PLACE SET
3. and a PLAN of transactions.
4. **Its main method** executes the plan leading to
   a) a **closed route around the all members** of the place set or
   b) a set of opened routes and unused places.

II. **Inherited properties and methods:** from FLAGS object.

III. **They have just like as other objects**

- Procedures supporting their creation and modification
  a) from **batch** input files
  b) or **interactive** dialogs.

IV. **Initial state:**

A. **Solution from empty state**

1. Set of unconnected places and
2. The method of distance calculation

B. **Continuing solution**

1. Remaining unconnected places
2. Calculated distances
3. Routes found in the continued solutions
4. Transaction log of the continued solution.

V. **Optional Precautions:**

A. Parameters for distance validation

1. Minimal accepted distance
2. Maximal accepted distance
3. Ratio of too far neighbors.

Extra Conditions and Results of the SOLUTION Objects

I. **User-defined edges**

1. Optional beginning place.
2. Optional fixed and prohibited edges.

II. **Set of the examined places**

a) Default: The members of the whole place set are examined.

b) Restricted: new places can be added to the existing routes from inside or outside of a region defined by its central place and a distance threshold.

III. **Selection of the best solution**

a) Default: Deterministic. The smallest added distance and the alphabetical order of the places determines the added edges.

b) Fuzzy: If more than one added edges result the same increase of the total length then the program randomly selects from them.

i. Parameters of random correction of distances enabling not to consume the short inadvertently.

IV. Calculated data structures

1. The triangular distance matrix

2. Routes (opened or closed, route index, total length, sequence of place indexes)

3. Transaction log.

Union

The UNION objects contain one or two solutions displayed in the same figure and the derived properties used as limits of displayed range of coordinates.

Properties

- They determine the limits of displayed ranges using the Place Sets referenced in their Used Solutions properties.
- The number of coordinates and the coordinate system must be common for them.
- The unified boundary values are counted for them.
- The recommended or user defined transformation are determined based on the applied view.
- The transformation of the longitudes are executed using the merged list of the sorted longitudes.

REPORT Objects

I. Identification and Sources

The results of the SOLUTIONS are finally tabulated by the Report objects.

I. Its own properties

1. Device symbol
2. Mode of output
   A. Separated
   B. Merged tables.
3. Unique file name or Window title of document, respectively
4. Mode of the report:
   a) Single report = tabulated results of a single solution
   b) Comparison = tabulated results of two solutions
5. Source(s) of the report (one for single report, two for comparison:
   1. The identifier(s) PLACE SET from which
   2. the reported SOLUTION was made.

REPORT Objects

II. Tabulating Methods

The results of solutions are reported in any or all of the tables below:

1. Source PLACE SET
2. Distance matrix
3. Final route(s) and their total length(s)
4. Summary of allowed and executed number of TRANSACTIONS.
5. Fate of edges: order of joining and cutting the edges between pairs of places

Each table has a generating method.

Report Objects

III. Inherited Properties and General Methods

A. From Flags object.

B. They have just like as other objects

- ❑ Procedures supporting their creation and modification
  - a) from **batch** input files
  - b) or **interactive** dialogs.

IV. Subordinate objects

A. **Message_Form_NF** windows object holding
the colored messages supported by the following objects:

B. **Colored_Messages** form object editing the text styles.

C. **Define_Styles** object calling the

D. **SciLex** library procedures.

FIGURE Objects

I. Identification and Sources

The results of the solution are finally plotted by the Figure objects.

I. Its own properties

1. Device symbol
2. Automatically generated unique identifier
3. File name of the document derived from its identifier
4. Mode of the figure:
   - a) Single figure = plotted results of a single solution
   - b) Comparison = tabulated results of two solutions
5. **Source(s) of the report** (one for single report, two for comparison:
   - i. The UNION of the plotted SOLUTIONS was made.
   - ii. Used Settings
   - iii. Used View

FIGURE Objects

II. Inherited Properties and General Methods

A. **Inherited properties**
   - From FLAGS object

B. **This object has just like as other objects** procedures supporting their creation and modification
   a) from batch input files
   b) and interactive dialogs

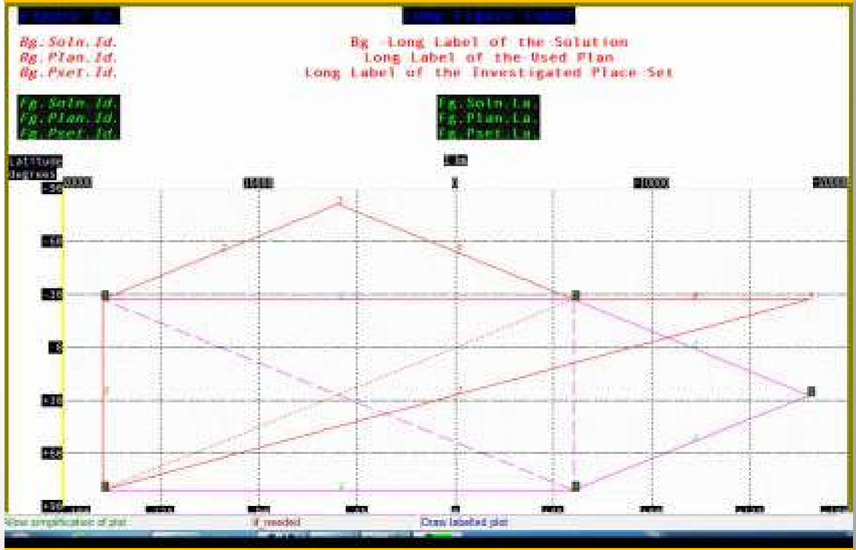
FIGURE Objects

III. Displayed Information

The figures consist of rectangles containing:

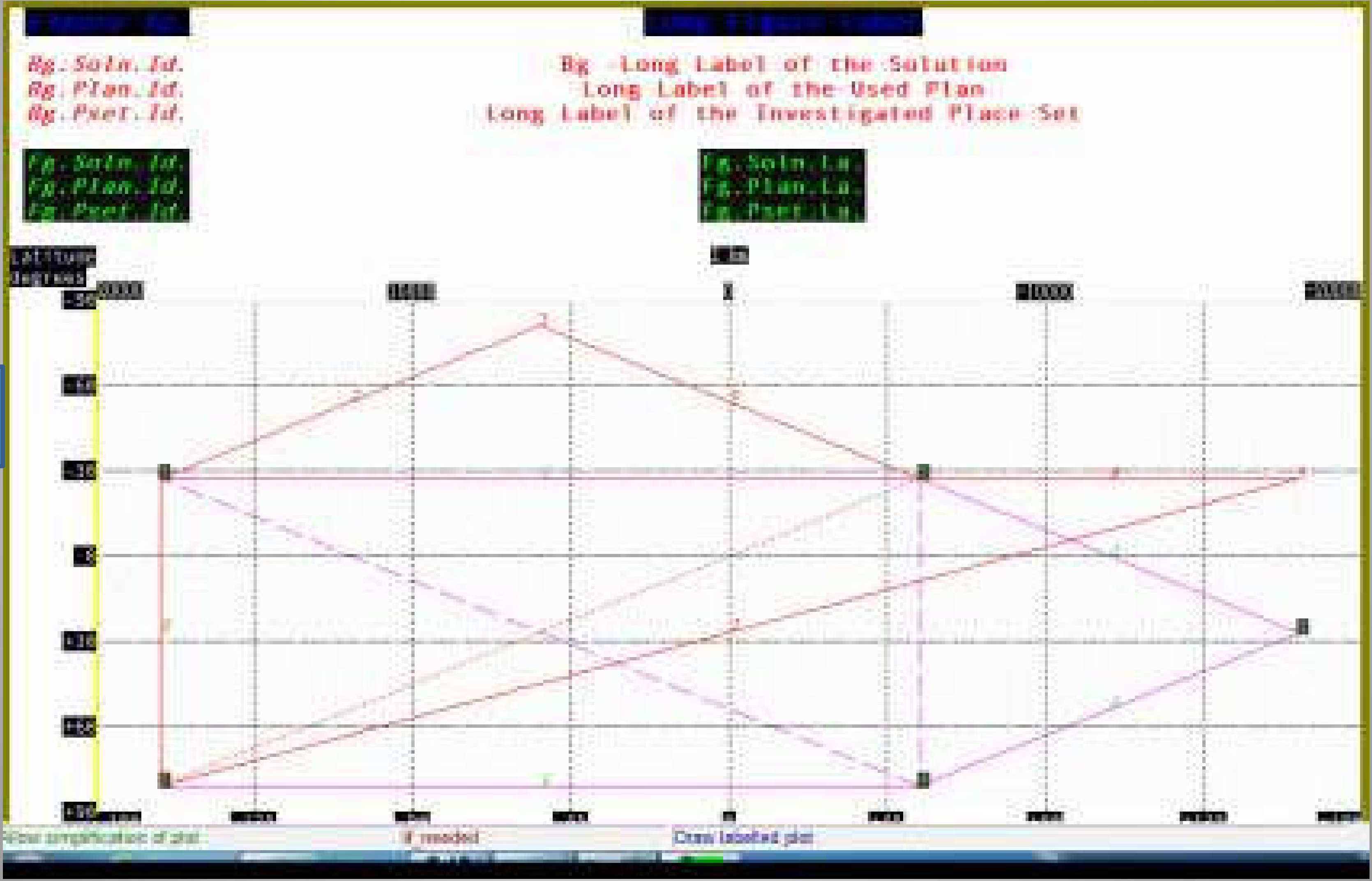
1. **Titles** of solution, plan and place set
2. **Labels** of them
3. **Axes** -- name, unit and scale
4. **Gridlines**
5. The static plot of the marked places
6. Plot of the **final route**(s)
   a) edges drawn in *one step*
   b) with a given *delay* in order of their appearance.

Sample plot made by the **TestDraw** program



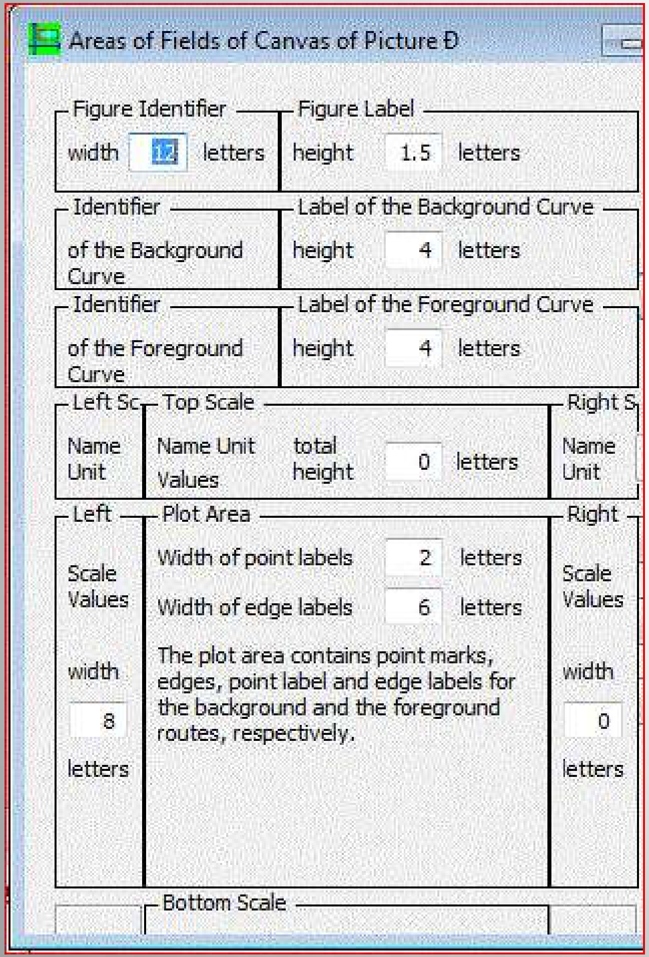
Click on figure in order to enlarge it.

Test Plot


Bg.Soln.Id.
Bg.Plan.Id.
Bg.Pset.Id.
Bg Long Label of the Solution
Long Label of the Used Plan
Long Label of the Investigated Place Set
Fg.Soln.Id.
Fg.Plan.Id.
Fg.Pset.Id.
Fg.Soln.La.
Fg.Plan.La.
Fg.Pset.La.
Latitude
Degrees
Allow simplification of plot
if needed
Draw labelled plot

Subordinate Objects of Figure Objects

I. SETTINGS



Areas of Fields of Canvas of Picture Ð

Figure Identifier: width 12 letters

Figure Label: height 1.5 letters

Identifier of the Background Curve

Label of the Background Curve: height 4 letters

Identifier of the Foreground Curve

Label of the Foreground Curve: height 4 letters

Left Sc: Name Unit

Top Scale: Name Unit Values; total height 0 letters

Right S: Name Unit

Left: Scale Values; width 8 letters

Plot Area: Width of point labels 2 letters; Width of edge labels 6 letters

The plot area contains point marks, edges, point label and edge labels for the background and the foreground routes, respectively.

Right: Scale Values; width 0 letters

Bottom Scale

The colors, pen styles, fonts, and sizes of the above rectangles are read from the SETTINGS.

A. Rectangular areas of the figure

B. Color compositions

C. Font definitions and sample texts

D. Locations of the message window

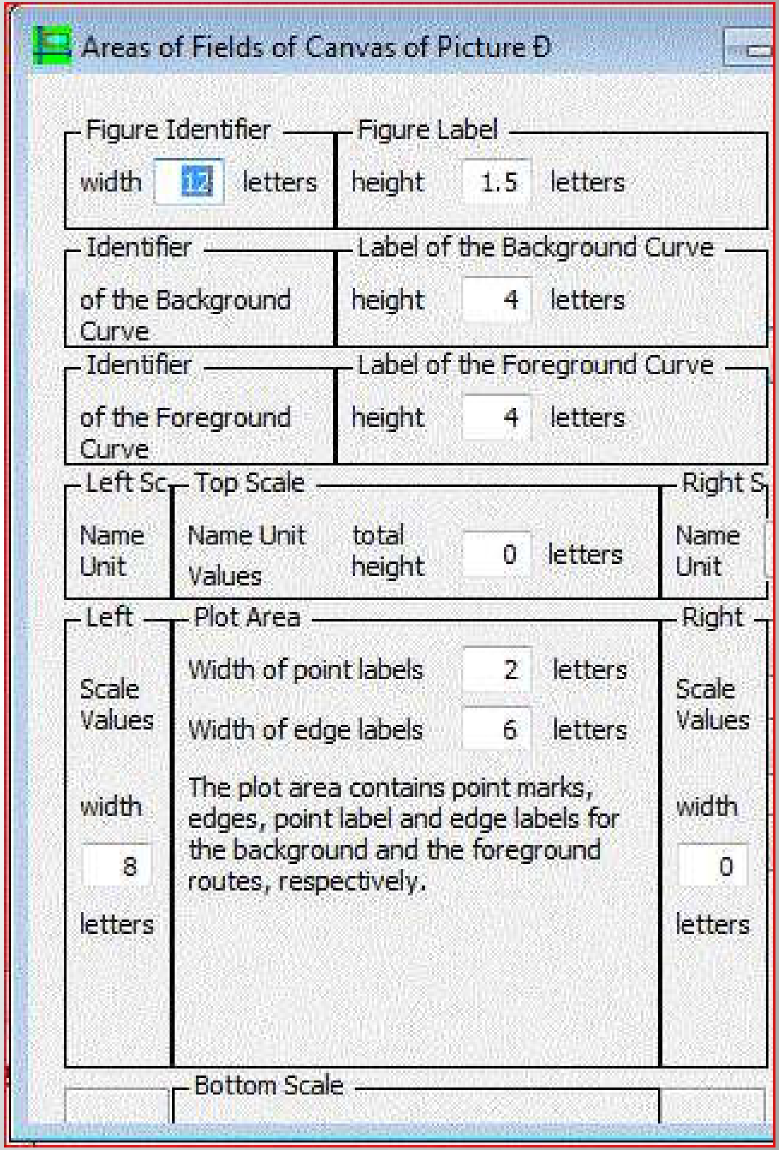
E. Pen styles

Rectangular Areas of the Figure

This dialog of the TestDraw program contains the sketch of the rectangular area of the figures.

The sizing of the rectangles is based on the **fixed 1000×1000 pixels of the inner plot area** and the sizes of the surrounding scales calculated from their heights and widths and fonts.

The „Đ” character in its title means that he default values are shown.



Areas of Fields of Canvas of Picture Đ

Figure Identifier: width 12 letters

Figure Label: height 1.5 letters

Identifier of the Background Curve

Label of the Background Curve: height 4 letters

Identifier of the Foreground Curve

Label of the Foreground Curve: height 4 letters

Left Sc: Name Unit

Top Scale: Name Unit Values; total height 0 letters

Right S: Name Unit

Left: Scale Values; width 8 letters

Plot Area: Width of point labels 2 letters; Width of edge labels 6 letters

The plot area contains point marks, edges, point label and edge labels for the background and the foreground routes, respectively.

Right: Scale Values; width 0 letters

Bottom Scale

Subordinate Objects of Figure Objects

II. View

I. **Inherited properties**

- From Flags object

II. **This object has just like as other objects** procedures supporting their creation and modification

1. from batch input files
2. and interactive dialogs

III. Own properties

A. **The limits of clip area** of shown places for each coordinates:

a) *User-defined* given as the center and extent of shown coordinates

b) *Fitted* to the range of displayed place set(s)

B. **The projection methods**

1. Map projections of the surface of the sphere to the plane of the map
2. Collineations of 3-dimensional data into 2 dimensions

C. The curved routes and gridlines over the sphere are drawn instead of arcs as polylines. *The number of their steps are defined by a built-in constant accuracy parameter, the maximal angle belonging to a segment*.

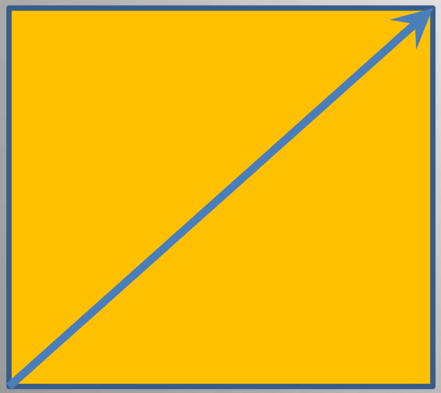
Chapter 2

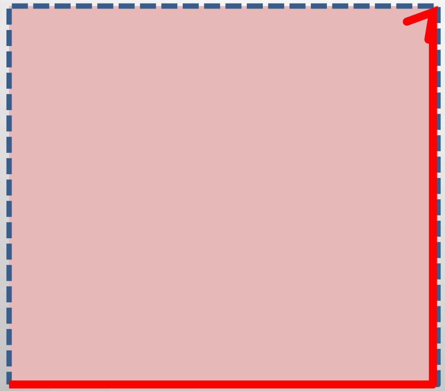
ALGORITHMS

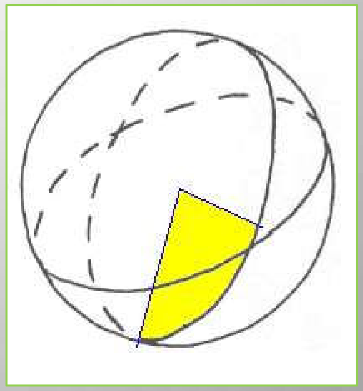
Next Chapter

Two-Dimensional Distance Formulae

Method	Variables	Definition	Equation or procedure
Eucledian	$P(x_1; y_1)$; $P(x_2, y_2)$	Cartesian coordinates	$d=[(x_1-x_2)^2+(y_1-y_2)^2]^{1/2}$
Manhattan			$d=\lvert x_1-x_2\rvert+\lvert y_1-y_2\rvert$
Great-circle	φ_1, φ_2 λ_1, λ_2 R $\Delta\varphi=\varphi_2- \varphi_1$ $\Delta\lambda=\lambda_2- \lambda_1$	Latitude Longitude Radius	$a = \sin^2(\Delta\varphi/2) + \cos(\varphi_1).\cos(\varphi_2).\sin^2(\Delta\lambda/2)$ $c = 2.\mathrm{atan2}(\sqrt{a}, \sqrt{(1-a)})$ $d = R.c$



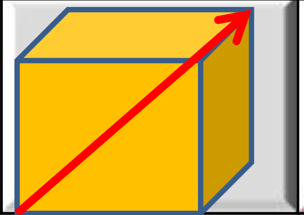




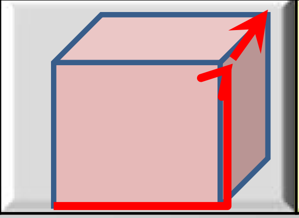
Distance Formulae

System	Method	Combination of shperical and radial distances	Variables	Definition	Equation or procedure
Cartesian	Eucledian		$P(x_1; y_1;z_1)$; $P(x_2, y_2;z_2))$	Cartesian coordinates	$d=[(x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2]^{1/2}$
	Manhattan				$d=\lvert x_1-x_2\rvert+\lvert y_1-y_2\rvert+\lvert z_1-z_2\rvert$
Spherical	Great-circle		$P(\varphi_1, \lambda_1, R_1)$; $P(\varphi_2, \lambda_2, R_2)$ $0 \leq R_1 \leq R_2$	Longitude Latitude Radius	$a = \sin^2(\Delta\varphi/2) + \cos(\varphi_1).\cos(\varphi_2).\sin^2(\Delta\lambda/2)$ $c = 2.\mathrm{atan2}(a^{1/2}, (1-a)^{1/2})$
		Eucledian			$d = [(R_1.c)^2+(R_2-R_1)^2]^{1/2}$
		Manhattan			$d = \lvert R_1.c\rvert+\lvert R_2-R_1\rvert$

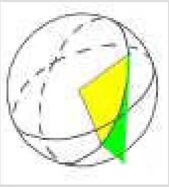
Eucledian



Manhattan



Great-circle



Classification of Viewed Subset of the Place Sets Given by Spherical Coordinates

- **The coarse distribution of their latitudes and radial distances** is used to recommend, allow and reject their mapping and projection methods.
- **The possible range of these coordinates are split into** *parallel belts.*
- **The algorithm determining the best mapping uses**
  1) *the places filtered by the views* according to their each coordinates,
  2) which *belts are empty or populated*, respectively.

I. *Latitude* Zones

Singular	Belt			Is populated? (Boolean)	Range of the latitude φ	Values of ε and ψ are given on the page of decision constants.
	Ordinal number used on next page	Hemi-sphere	Zone			
YES	+4	Northern	polar	P_N	$\varphi \geq 90^\circ-\varepsilon$	
	−4	Southern	polar	P_S	$\varphi \leq -(90^\circ-\varepsilon)$	
NO	+3	Northern	high	H_N	$\psi \leq \varphi < 90^\circ-\varepsilon$	
	−3	Southern	high	H_S	$-\psi \leq \varphi <-(90^\circ-\varepsilon)$	
	+2	Northern	low	L_N	$\varepsilon \leq \varphi < \psi$	
	−2	Southern	low	L_S	$-\varepsilon \leq \varphi < \psi$	
	+1	Northern	equatorial	Q_N	$0 \leq \varphi < \varepsilon$	
	−1	Southern	equatorial	Q_s	$-\varepsilon < \varphi < 0$	
May be	[+1, +4]	Northern	total	$T_N=P_N \vee H_N \vee L_N \vee Q_N$		
	[−4, −1]	Southern	total	$T_S=P_S \vee H_S \vee L_S \vee Q_S$		

II. Zones of Relative Distortion of Parallels

The ordinal numbers defined on the previous page of the lowest and the highest populated latitude zone determine the zones of the relative distortion.

–4	–3	–2	–1	+1	+2	+3	+4	Highest Lowest
Unaccept-able	U	U	U	U	U	U	U	–4
	Chk#1	Chk#1	Chk#1	Chk#1	Chk#1	Chk#1	U	–3
		Chk#1	Chk#1	Chk#1	Chk#1	Chk#1	U	–2
			Lowly	Lowly	Chk#1	Chk#1	U	–1
				Lowly	Chk#1	Chk#1	U	+1
					Chk#1	Chk#1	U	+2
						Chk#1	U	+3
							U	+4

<u>Chk#1:</u>
Determine range of cos(latitude) for all places in view.

$$\max(\cos\varphi) - \min(\cos\varphi) \begin{cases} < \sigma & \text{lowly} \\ \geq \tau & \text{very} \\ \textit{else} & \text{moderately} \end{cases}$$

Discarding of the Singular Places

The singular places are not used in the algorithm of the determination of the best central meridian.

A place given in spherical coordinates is singular if

1. its latitude is the neighborhood of the poles or
2. it is too close to the center point.

The corresponding formulae are

1. $|\varphi| \geq 90° - \varepsilon$ and
2. $\rho \leq \zeta$, respectively.

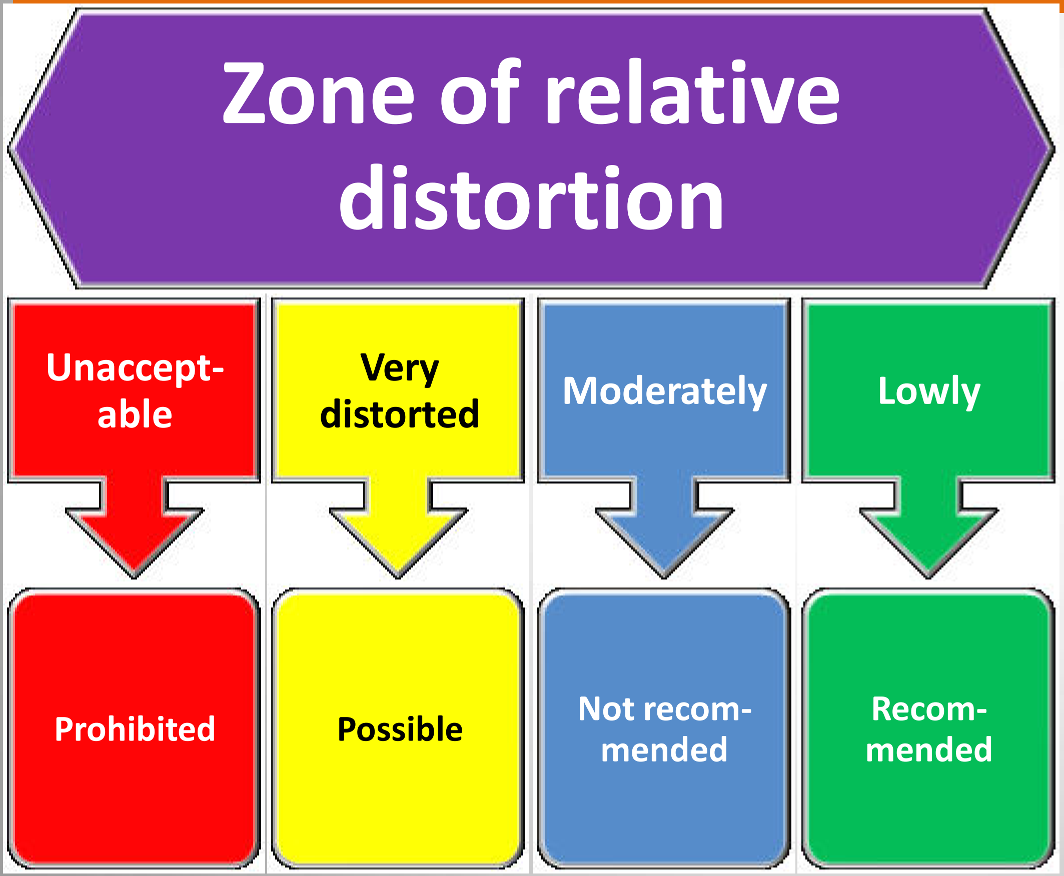
Here φ is the latitude, and ρ is the radius coordinate; ε and ζ are decision constants.

Decision Constants for the Mapping of Spherical Coordinates

Symbol	Explanation	Value and unit	Used in
ε	Maximal latitude difference from the poles and the equator, respectively	0.01°	Classification of places by their latitudes
ψ	Lower limit of the high latitudes	70.0°	
ζ	Threshold for central singular points in case of three-dimensional spherical coordinates	0.01 km	
σ	Lower limit of the moderate distortion of parallels	0.01	Permission of cylindrical transformation
τ	Lower limit of the very big distortion of parallels expressed as the diference of the cosine of latitudes	0.2	
β	Maximal angle belonging to a step drawn instead of the a segment of a gridline or a route over the sphere.	$\pi/32$ radian that is 5.625°	Fine drawing of curves

Choosing of the Recommended Map Projections of Sphere to Plane *I. Cylindrical Projection*

These projections map the surface of the sphere to another surface - e. g. a plane, the lateral surface of a cylinder or a cone - which can be unfolded to a plane.



Zone of relative distortion

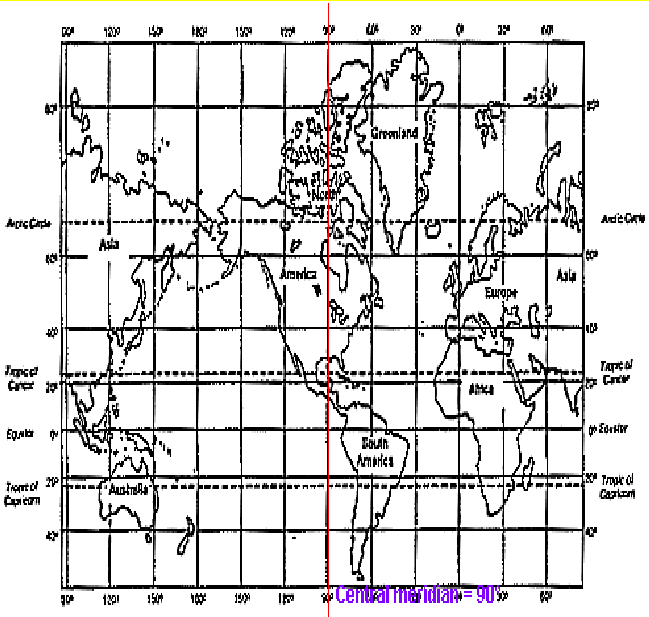
Unaccept-able → Prohibited

Very distorted → Possible

Moderately → Not recom-mended

Lowly → Recom-mended

Cylindrical projection with American central meridian



This projection stretches extremely the polar parallels.

Read more about projections from the external document *Projection of Data on the Figures.docx*

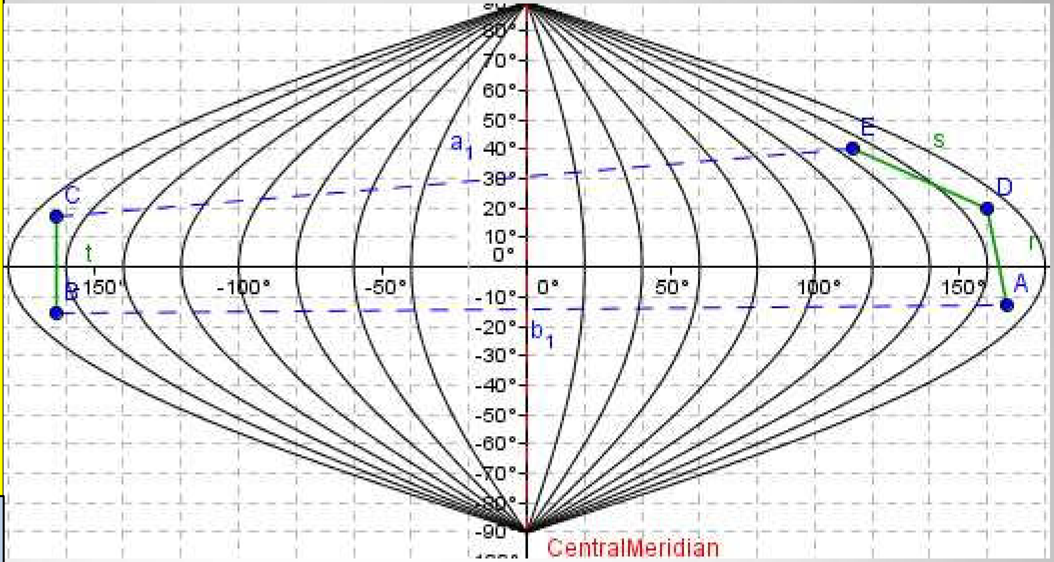
The Central Meridian

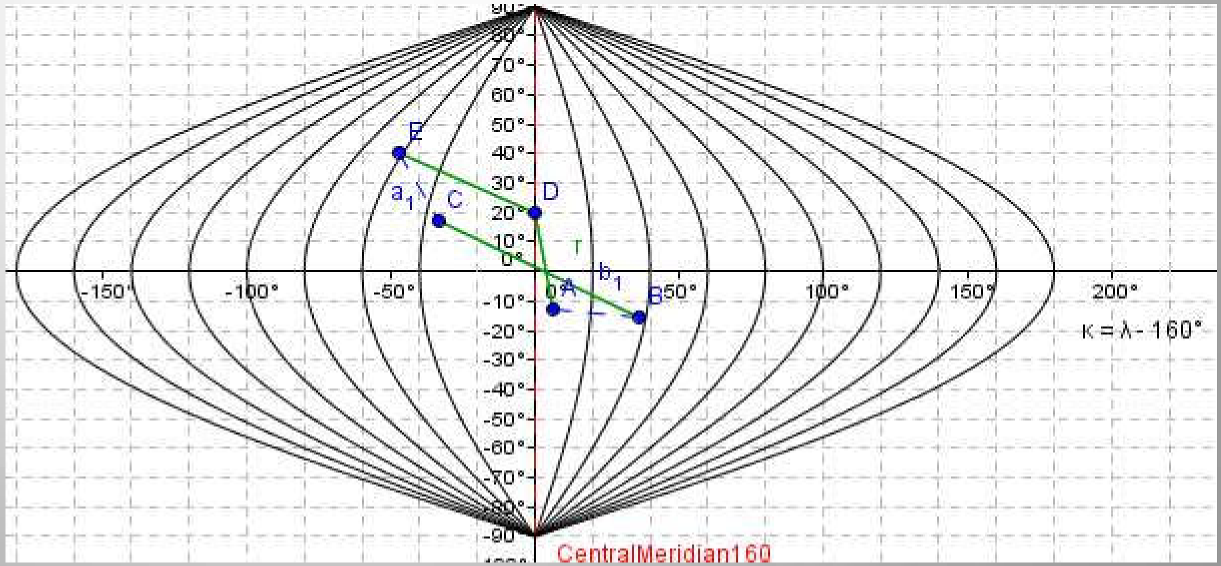
The central meridian is the straight meridian in the middle of the projection.

If it is badly chosen then the shown distances are very distorted and edges appear leading to left frame to the right frame .

The best C central meridian is that when

- ✓ *the displayed places and routes remain in the least distorted middle of the figure and*
- ✓ *the resulting extent covering all κ transformed longitudes is minimal .*

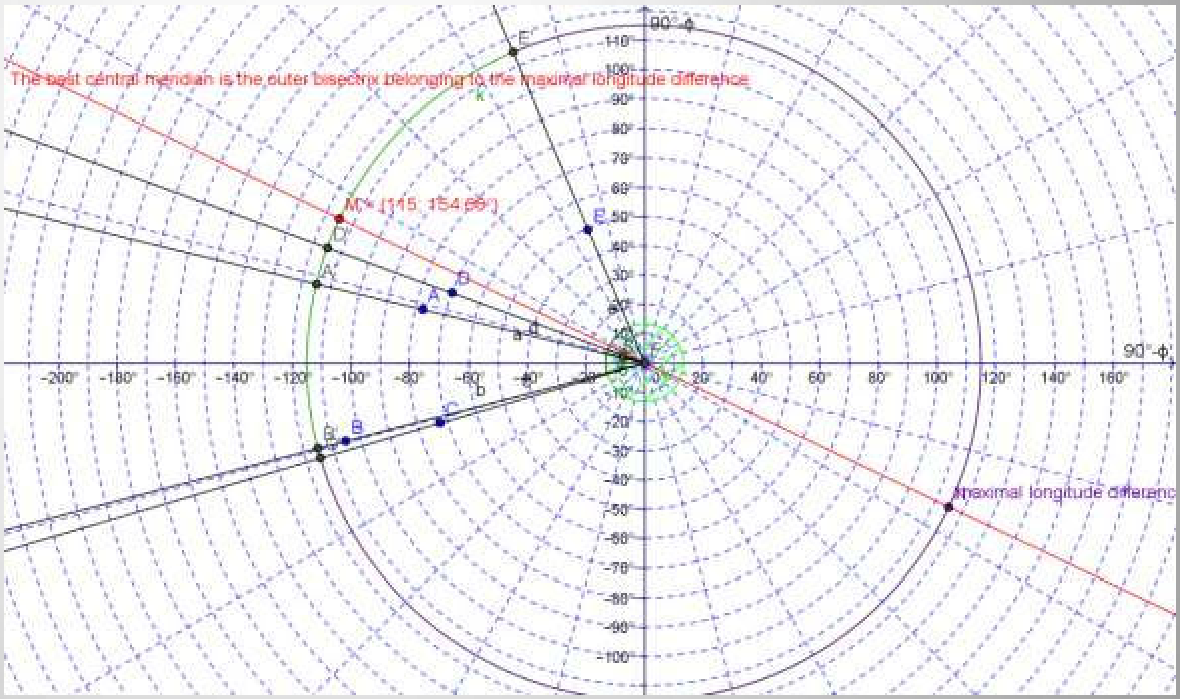




Determination of the Best Central Meridian

1. Gather the longitudes of the non-singular places.
2. Sort the latitudes in increasing order.
3. Discard duplicated values.
4. Add 360° to the lowest value and append the result to the end of the list.
5. Look for the maximal difference of neighboring elements.
6. The outer bisectrix of the found angle will be the best central meridian.

The best central meridian is the outer bisectrix belonging to the maximal longitude difference.



Read more about projections from the external document *Projection of Data on the Figures.docx*

II. Transformation to Cartesian XYZ Coordinates

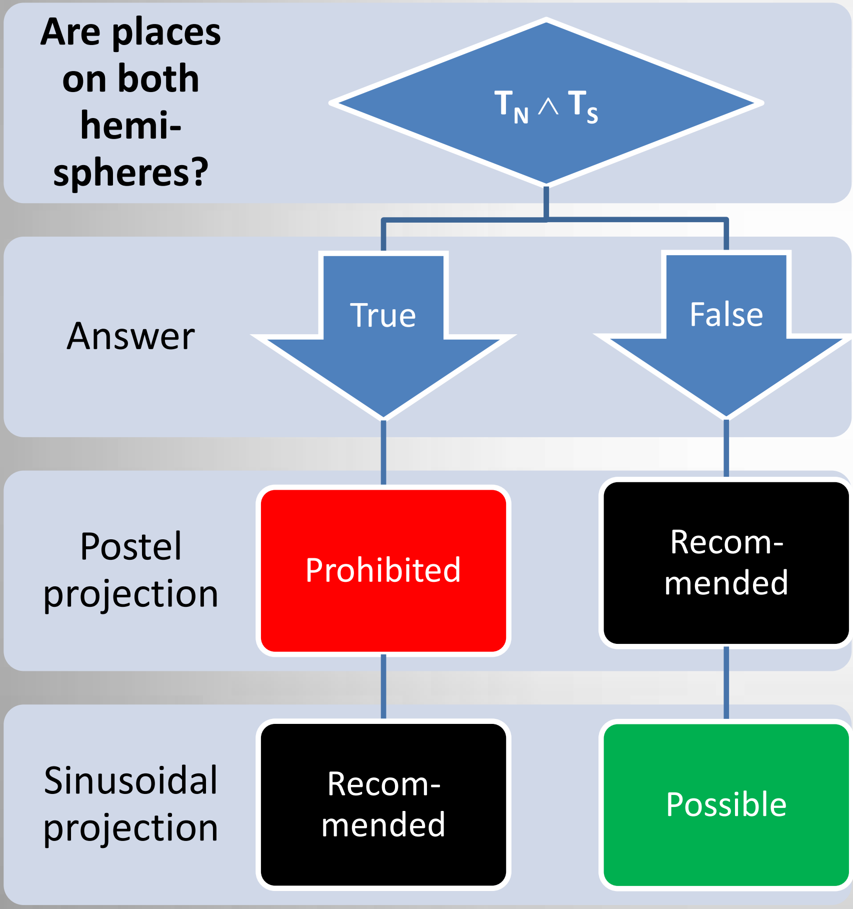
The three-dimensional spherical coordinates may be transformed to Cartesian XYZ coordinates.

In Visual Prolog notation

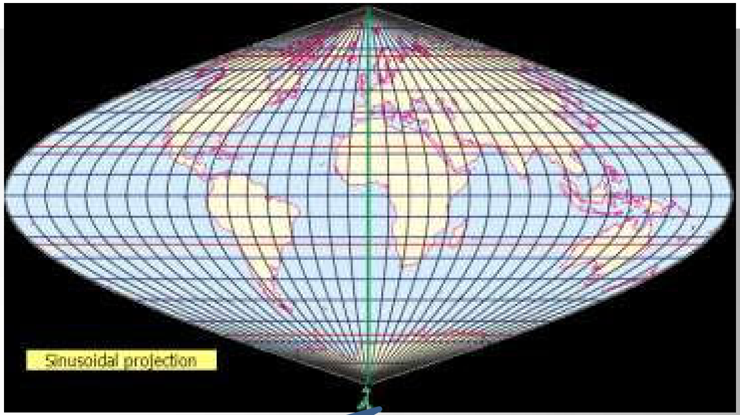
```
getX()   =X :-
        coordinates=s3(Latitude,Longitude,Radius),
        X=Radius*cosd(Latitude)*cosd(Longitude),
        !.
getY()   =Y :-
        coordinates=s3(Latitude,Longitude,Radius),
        Y=Radius*cosd(Latitude)*sind(Longitude),
        !.
getZ()   =Z :-
        coordinates=s3(Latitude,_Longitude,Radius),
        Z=Radius*sind(Latitude),
        !.
```

Read more about projections from the external document
Projection of Data on the Figures.docx

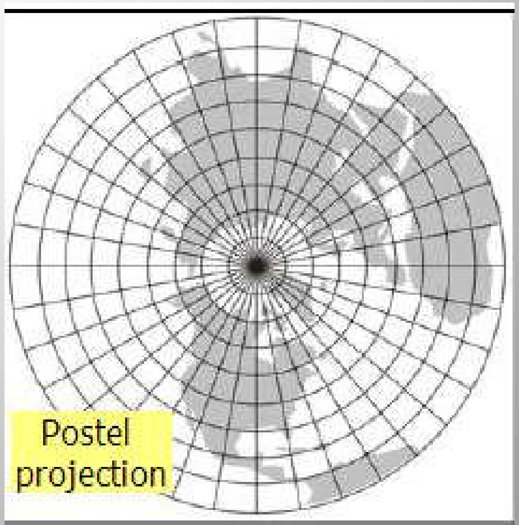
III-IV. Postel and Sinusoidal



Are places on both hemispheres?	$T_N \wedge T_S$	
Answer	True	False
Postel projection	Prohibited	Recommended
Sinusoidal projection	Recommended	Possible

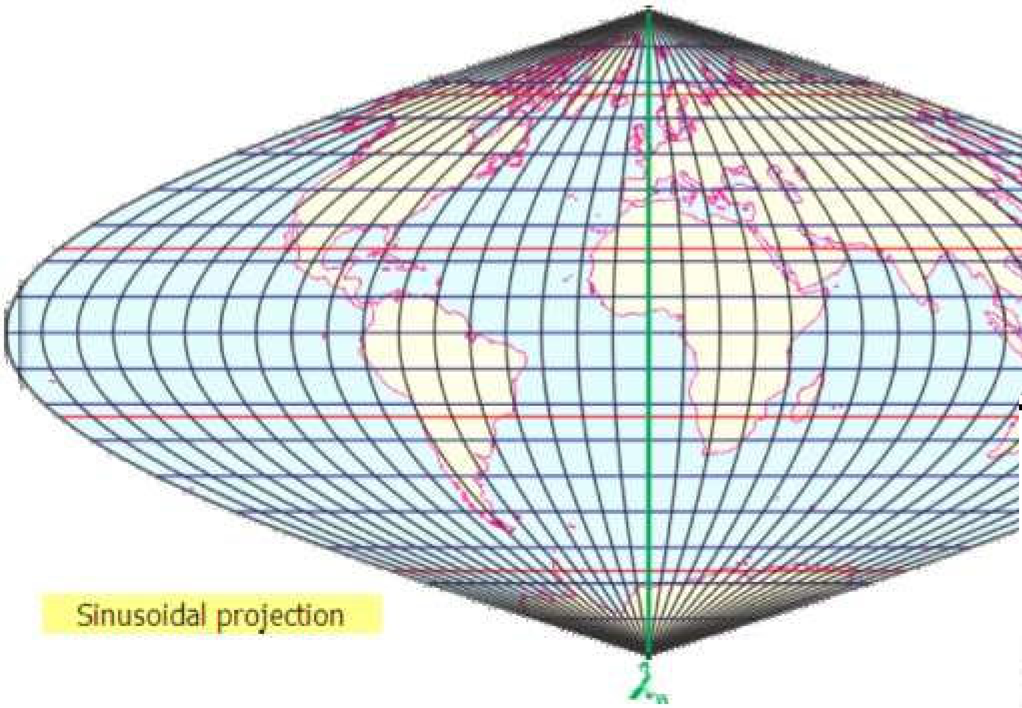


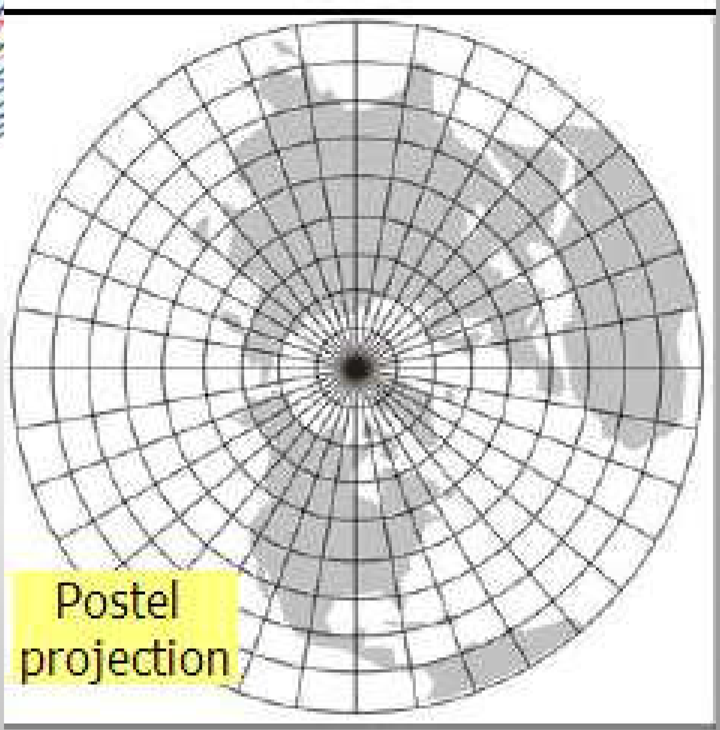
Central meridian



Read more about projections from the external document *Projection of Data on the Figures.docx*

Enlarged Comparison of Projections





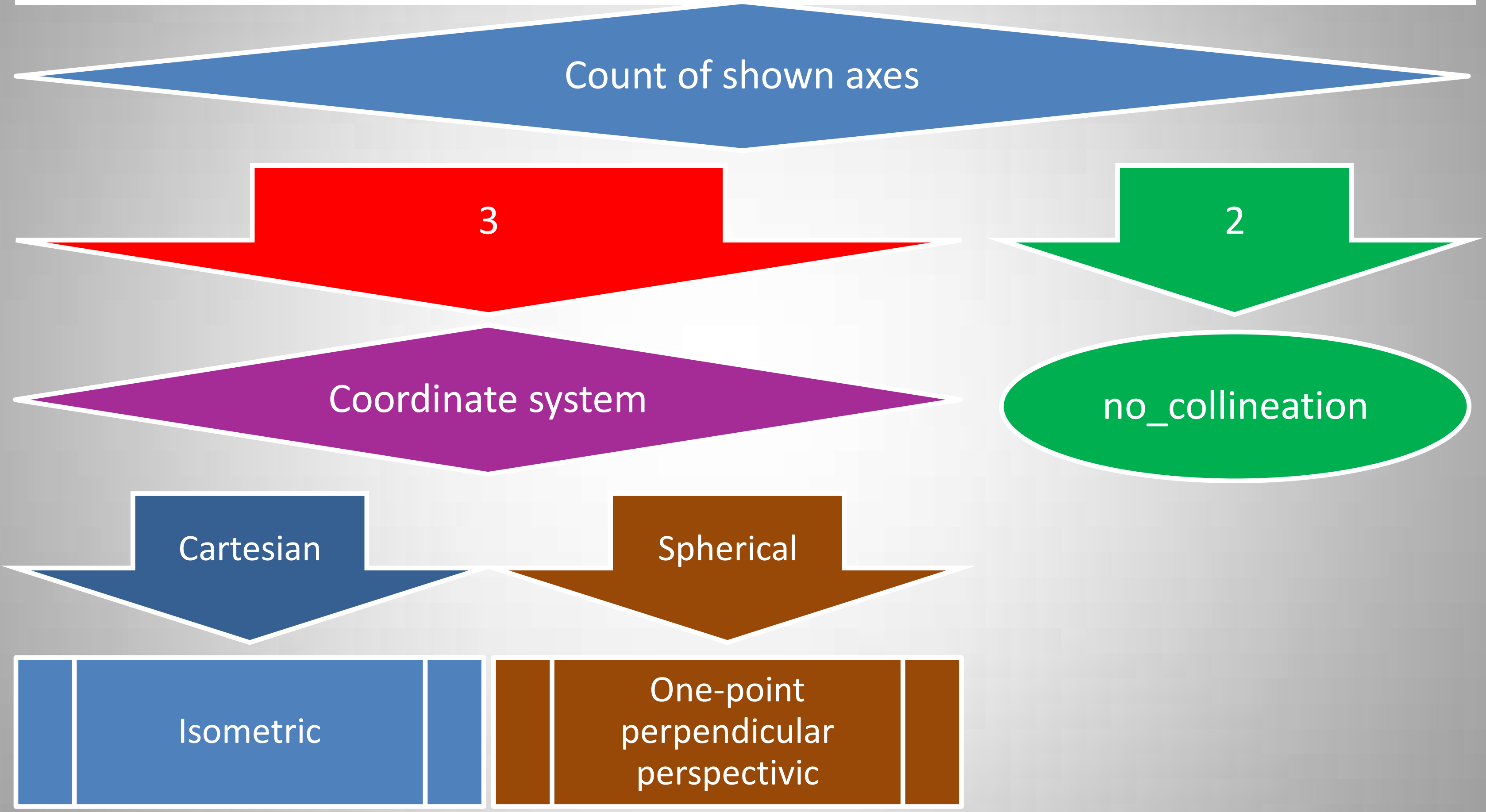
- *The sinusoidal projection distorts the areas far from the central meridian.*
- *The Postel projection distorts the belts of the opposite hemisphere.*

Handling of Three-Dimensional Data

A. Omitting one of the coordinates from the figure
   a) If latitudes and longitudes remain then they need map projections of the surface of the sphere to the plane of the map,
   b) otherwise use them as horizontal and vertical coordinates of the view.

B. Projecting them in one or two steps
   a) The three Cartesian coordinates need only one more step , the collineation of three-dimensional data into two dimensions.
   b) The three-dimensional spherical coordinates need two steps:
      1. A map projection of the spherical coordinates to Cartesian one, namely
         A. transformation of latitudes and longitudes to u and v resulting coordinates, leaving the ρ radial distance unchanged;
         B. Transformation of all three spherical coordinates to x, y and z Cartesian coordinates.
      2. a collineation the (u, v, ρ) , (x, y, z) resulting coordinates , respectively to (ξ, η) value pairs.

Recommended Method of Collineation of Place Sets

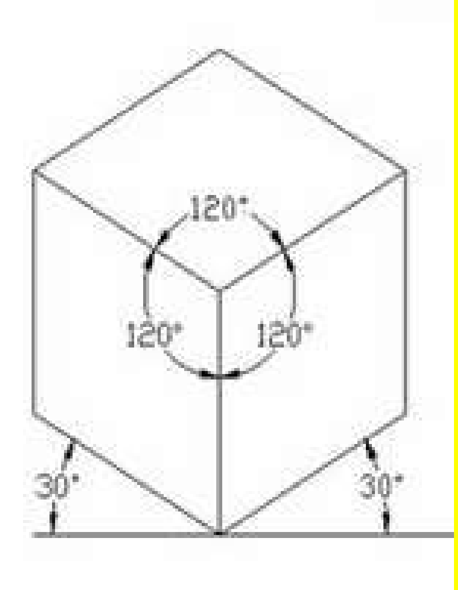


Read more about projections from the external document *Projection of Data on the Figures.docx*

Available Kinds of Collineations of Three-Dimensional Data

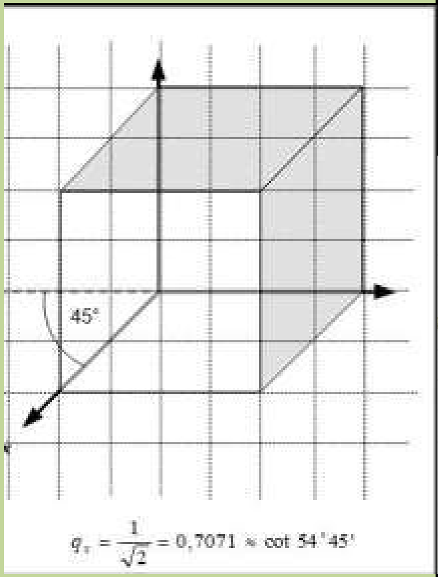
A) Isometric axonometric

The projection of a body diagonal is a point.



B) Modified Cavalier axonometric

The shrinking in the X direction is $q_x = \frac{\sqrt{2}}{2} = 0.7071$



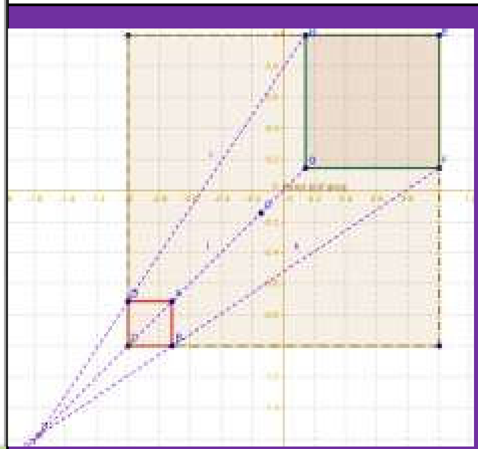
The ratio of areas of the opposite vertices is 1:1.

One-point perspective

The projections of the squares {Z=+1, |X|≤1, |Y|≤1} and {Z=-1, |X|≤1, |Y|≤1}

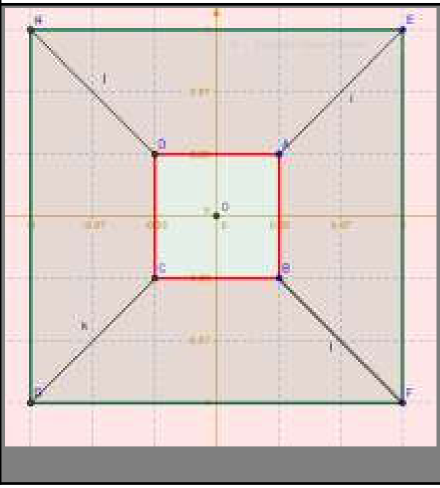
C) Rotated

do not overlap.



D) Perpendicular

do overlap.



The ratio of areas of the above squares is 1:9.

Substitution of Arcs of Great-Circles by Polylines

In case of two-dimensional spherical coordinates the routes between connected places are arcs of great-circles of the sphere. *This version of program does not display the arcs as polylines in order to simplify the plotting algoritm.*

If the connected places are antipodal that is

$$|\varphi_1+\varphi_2| < \varepsilon \text{ and } 180° - \varepsilon < |\lambda_1-\lambda_2| = 180° + \varepsilon$$

then the route is drawn using an third point $(\varphi_{3,}\ \lambda_3)$ where

$$\lambda_3 = \lambda_1$$

and

$$\text{if } |\varphi_1\,| \geq\ 90°-\varepsilon\ \text{ then } \varphi_3 = 0 \text{ else } \varphi_3 = 90°.$$

In this case the splitting detailed on the next page must be done for arcs

(φ_1,λ_1) to (φ_3,λ_3) and

(φ_3,λ_3) to (φ_2,λ_2), respectively.

Splitting of the Long Arcs

If the central angle *c* belonging to arc (φ_1,λ_1) to (φ_2,λ_3) is greater than the preset β constant then the arc is divided to $N = \text{int}\left(\frac{\beta}{c}\right)$ parts at factors $f_i = \frac{i\beta}{c}$

The indices, the factors and the coordinates of the separating points are

$$\begin{array}{lll} i = 0 & f_i = 0 & \hat{P}_0\left(\hat{\varphi}_0, \hat{\lambda}_0\right) = P_1\left(\varphi_1, \lambda_1\right) \\ i = 1, \ldots, N & f_i = \frac{i\beta}{c} & \hat{P}_i\left(\hat{\varphi}_i, \hat{\lambda}_i\right) \text{counted below} \\ i = N+1 & f_{N+1} = 1 & \hat{P}_{N+1}\left(\hat{\varphi}_{N+1}, \hat{\lambda}_{N+1}\right) = P_2\left(\varphi_2, \lambda_2\right) \end{array}$$

The internal cycle:

Let us substitute the above f_i values in place of f below and store its results φ and λ in the coordinates $\hat{\varphi}_i$ and $\hat{\lambda}_i$, respectively.

A=sin((1-f)*c)/sin(c)

B=sin(f*c)/sin(c)

x = A*cos(φ_1)*cos(λ_1) + B*cos(φ_2)*cos(λ_2)

y = A*cos(φ_1)*sin(λ_1) + B*cos(φ_2)*sin(λ_2)

z = A*sin(φ_1) + B*sin(φ_2)

φ=atan2(z,sqrt(x^2+y^2))

λ=atan2(y,x)

The calculation is taken from here. The symbol of variables are changed in order to match with this earlier page of the manual.

Validation of Places Checking Their Distances

This check helps the users to find the typos during the entering the coordinates of the places.

1) **Check of duplicated places:**
   If the distance of two or more places is smaller than a given limit L then probably only one of them is valid.

2) **Check of outsiders:**
   If a place is farther from others than a given limit H then its coordinates are probably invalid. If the t_i ratio of too high distances is greater than a given ratio T of the counted distances then the place is outsider.

$$\text{Calculate for } \forall i \in [1, p]$$

$$D(P_i, P_j) < L,\ j = 1, \ldots, p,\ j \neq i$$

$$\text{Count for } \forall\ i \in [1, p]$$

$$D(P_i, P_j) > H,\ j = 1, \ldots, p,\ j \neq i$$

$$\Rightarrow c_i$$

$$t_i = \frac{c_i}{p-1} > T > 0.5$$

Kinds of Plans

Kind	Essence
Full	Each $(p-1)!/2$ possible circular permutations of the p places of the whole place set are compared or each $(s-1)!/2$ possible circular permutations of the s places of its selected subset are compared.[1, 2]
Greedy	Take in account only one or very few steps of the solution in each transaction. It is name greedy because the premature consuming of certain short edges at the earlier stage of the iteration may lead to a suboptimal whole route. Each transaction returns the best added edge(s).[3, 4]
Undo	Open at one edge or fragment the single closed routes got from the continued greedy search. It makes possible to search for another solution in a subsequent a greedy search. [3, 4]

Remarks:

1 The full search is available only if the p *or* s *number of involved places does is less than the given threshold* m given in the interactive Solution Options dialog documented in *the help file of the itnerative data entry*.

2 The full search may not be continued if it extends to all places of the set.

3 Deterministic mode: If more equidistant place pairs are present then the alphabetical order of their identifiers determines the chosen edge(s).

4 Random mode: If more equidistant place pairs are present then the best result is chosen by adding random correction to the distances which provide the selection even from slightly longer edges.

Partial Solutions of Too Large Problems

Type of restriction	Cases	Mode of selection
Usage of some subsets of the solved place set.	Default: All places of the set.	----
	Optional: Pre-selected places.	Give set of first letters of the selected place names.
Prefix place used in the first transaction.	Default: Any place of the set usable in the allowed transactions.	----
	Optional: Pre-selected places.	Give a complete place name.
Preset state of some edges.	Default: No conditions.	---
	Edges which must be or must remain connected.	Give two place names.
	Edges which must cut or must not be connected.	Give two place names.

The Full Search

The circular permutations of the involved places are generated recursively.
The reversed routes are ignored:
The order of 3 places is [3,2,1].
The *K*-th place may be inserted in the route of *K*-1 places
before the first point of the route = [4, 3, 2, 1];
between any other places: [3, 4, 2, 1], [3, 2, 4, 1.
The inserted place is deleted from the list of available places
The process of insertion is continued until the list of available places becomes empty.
The sum of the length of edges is calculated and compared with the lowest sum.
If the current sum is smaller then the previous sum the route is stored.

Search	Trans-action	Repeat-able	Allowed when the count of places …	Explanation: Permute places circularly in order to find the best solutions.
Full	permute	No	… is smaller than a wired-in count of places is over a preset limit $p < m$.	**Compare <u>all possible</u> different closed routes.** The count of possible routes connecting p places is $t = (p–1)! /2$. The value of t can be extremely large. See *the factorial calculator here*.

Transactions of the Solutions

I. Necessary Simple Transactions

The repeatable greedy and undo transactions may use random corrections of the distances .

The formula of random correction is

$$\hat{d} = d(1+q\omega)$$

It uses a random number and a constant

$$q \in [0,1]$$

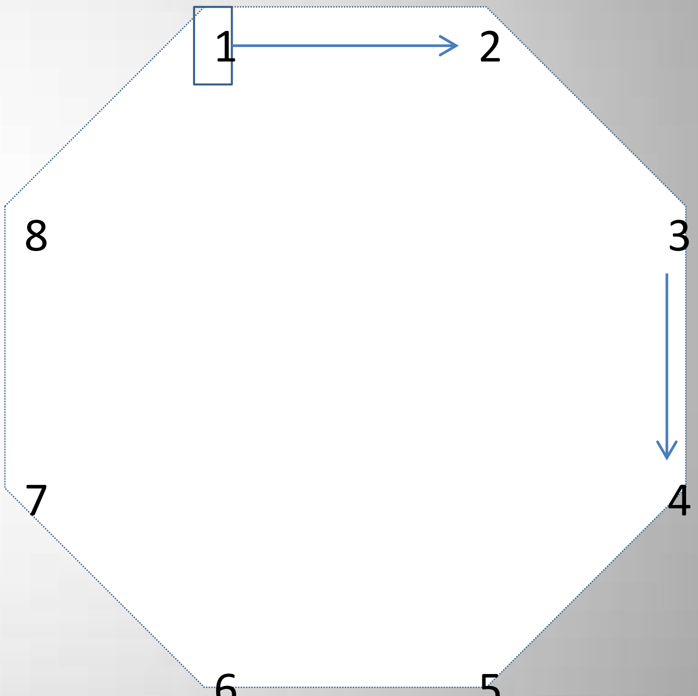
ω.

The table enlists the simple transactions required to build up a closed route from disconnected places adding the edges one by one.

Repeatable	**Greedy transactions**	**Explanation**
Yes	start	Start a new route from two places having no connections.
	continue	Continue a route with an already unused place.
	connect	Connect two routes at their head or final points.
No	close	Close the remaining single route when no unused places have been remained.

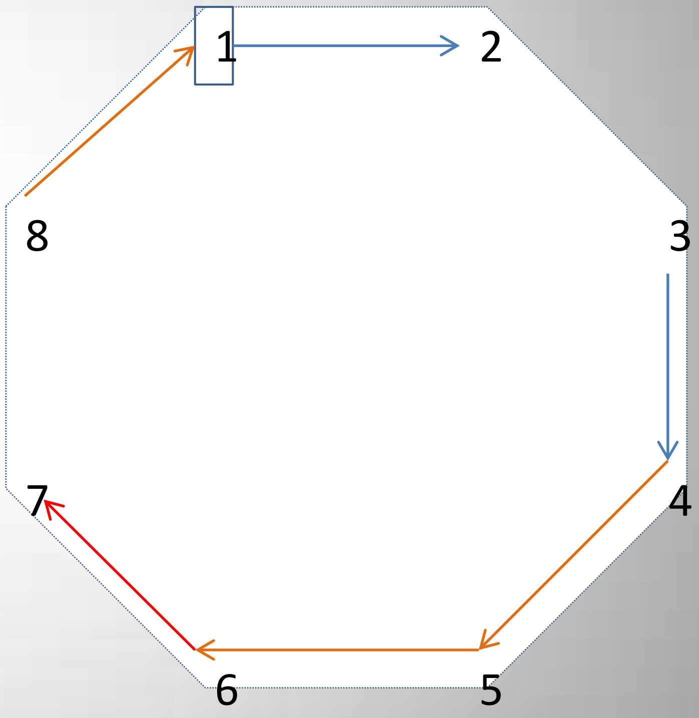
Line Diagram of Transaction START

Start a new route from two free places.



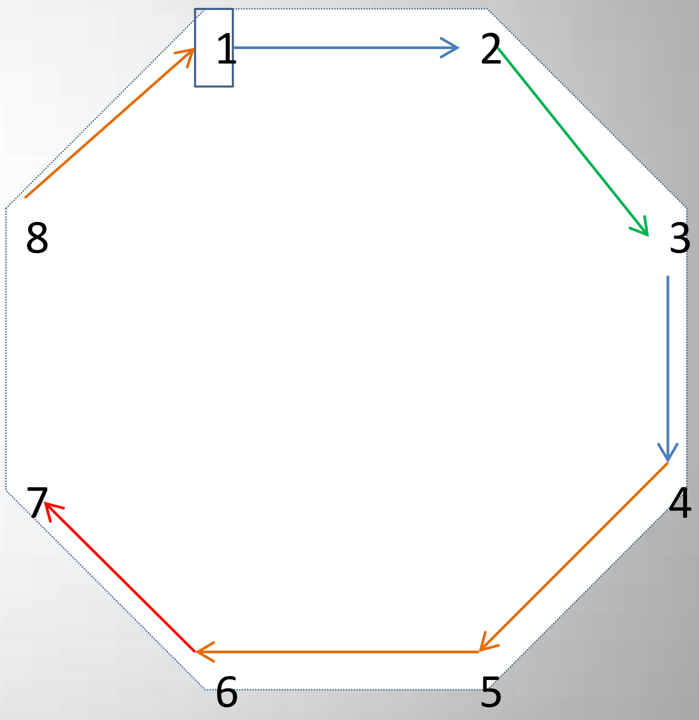
Line Diagram of Transaction CONTINUE

Continue a route with a free place.



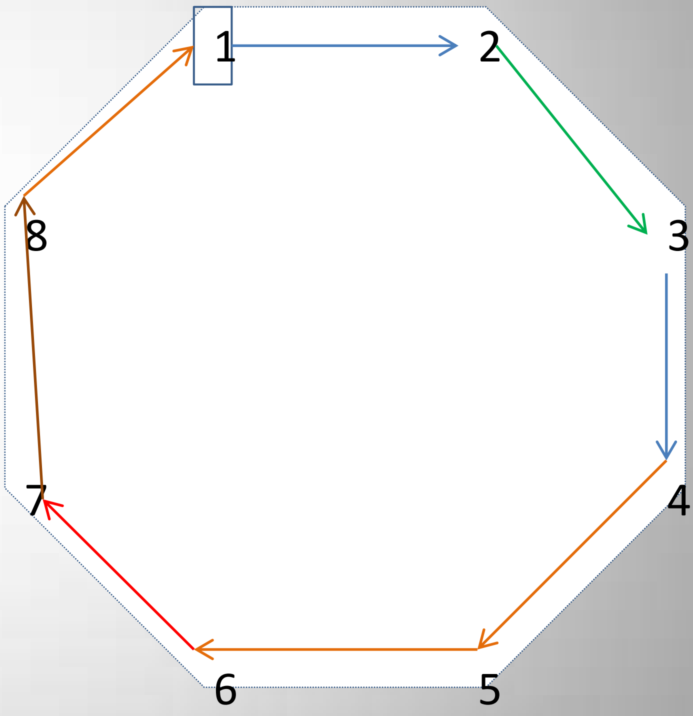
Line Diagram of Transaction Connect

Connect two routes at their head or final points.



Line Diagram of Transaction CLOSE

- Close the remaining single route when no unused places are present.
- This transaction can be executed at most once during a solution.
- It does not use randomly corrected distances.

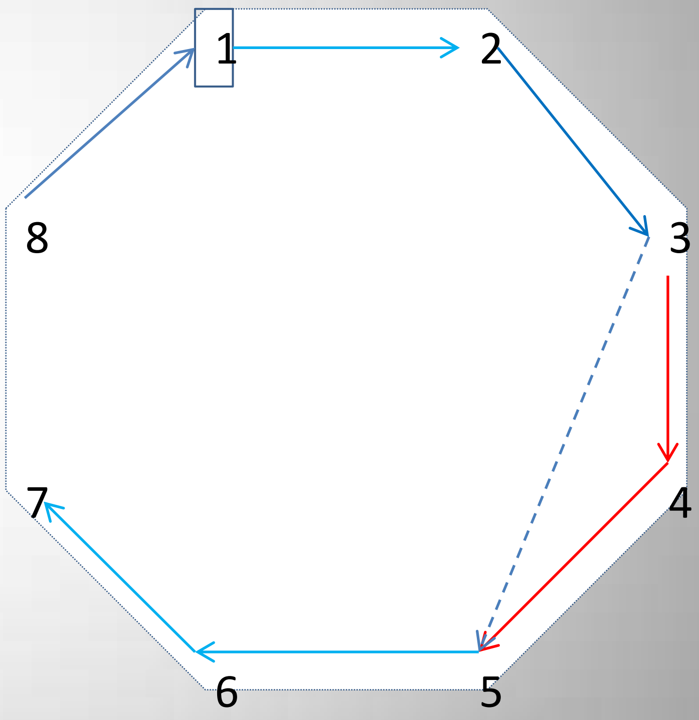


II. Sophisticated Transactions

Repeatable	State of output routes	Transaction	Edges		Input				Output	Process	
			Cut	Joined	Count of involved		Total count of		Count of resulting	Cutting step	Joining step
					Routes	Unused places	Routes	Unused places	Routes		
Yes	Open	insert	1	2	1	1	≥1	≥1	1	Cut a route somewhere.	Insert a single place between the new endpoints.
		exchange	2	4	2	0	≥2	≥0	2	Cut out a place from both routes.	Insert the cut places at the original position of the other cut place.
		reverse	2	2	1	0	≥1	≥1	1	Cut out a part of a route.	Insert back the cut part in reversed order.
		merge	1	2	2	0	≥2	≥0	2	Cut an edge of an unclosed route.	Insert a selected other whole route among the new endpoints.
		swap	2	2	2	0	≥2	≥0	2	Split two opened routes into two parts.	Join the fragments cut from different input in the best order.
		clamp	0	2	2	1	=2	=1	1	None	Clamp the two input routes across the selected place.
No	Closed	brace	2	2	2	0	≥2 ALL CLOSED	=0	1	Split two closed routes	Brace the got open routes at their both ends.

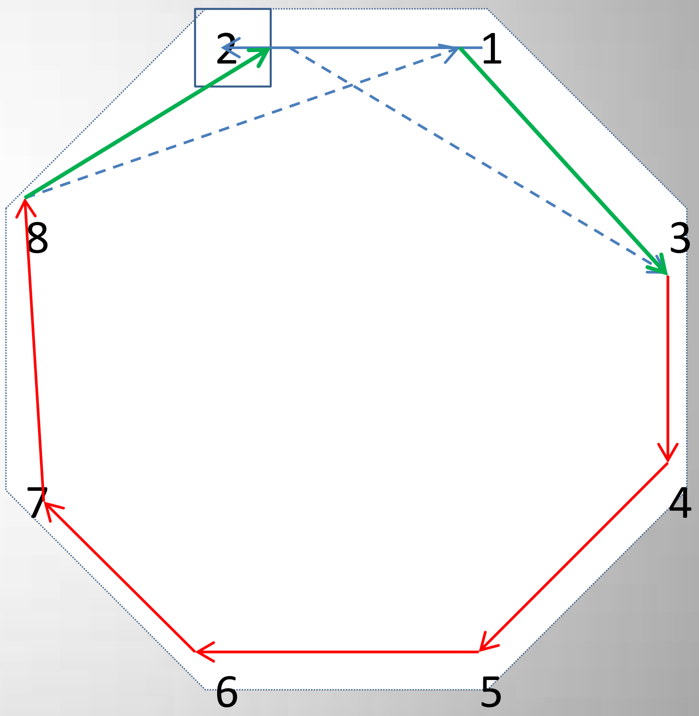
Line Diagram of Transaction INSERT

Insert a single place between two stations of the route.



Line Diagram of Transaction REVERSE

1. Cut a part of a route
2. *then insert it between the residual fragments in reversed order.*



Line Diagram of Transaction MERGE

Merge a whole route between two places of another route

Routes before the transaction:

[1,2,3,4] and [5,6,7,8]

Cut edge: 6 → 7.

A. in original order

Joined edges 6→1 and 4→7.

Route after the transaction:

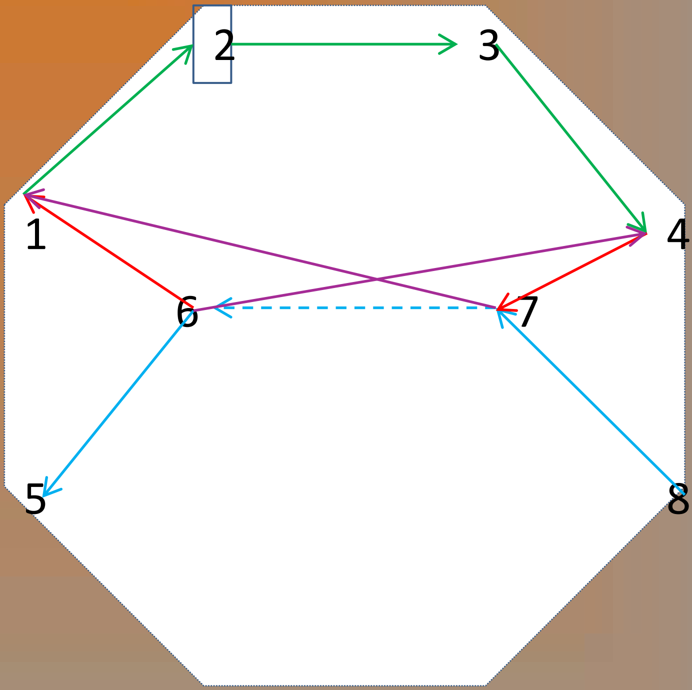
[5,6, 1,2,3,4 ,7,8].

B. in reversed order

Joined Edges: 7→1 and 4→7.

Route after the transaction:

[5,6, 4,3,2,1 ,7,8]



Line Diagram of Transaction SWAP

1. **Split two opened routes at an edge.**

2. **Join the fragments got from different edges.**

<u>Example</u>

0. Routes before the transaction:

R1=[11,12,13,14,15,16]

R2=[21,22,23,24,25,26]

1. Cut edges: 14 → 15 and 22→23.

Fragments on the example:

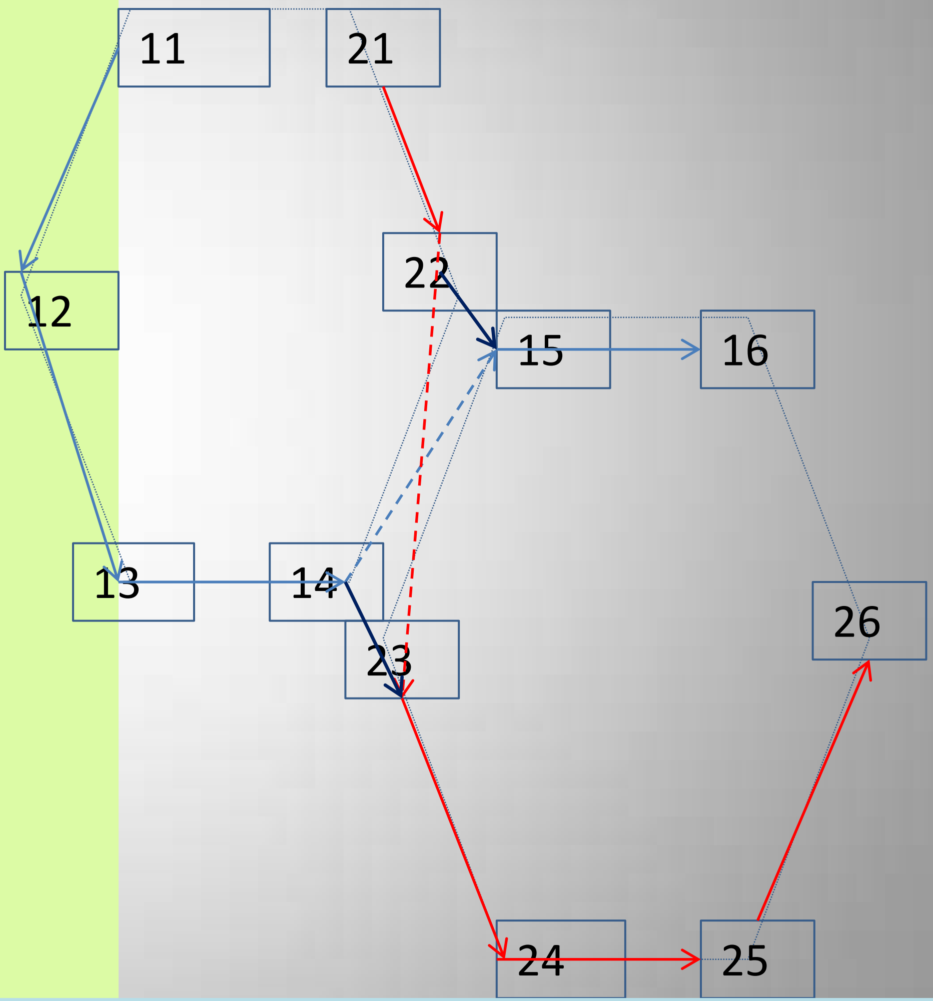
F11 = [11,12,13,14] and F12 = [15,16]

F21 = [21,22,23,24] and F22 = [25,26]

2. Possible pairs of swapped routes:

a. S1’=RO(F11)+RO(F21) and S2’=RO(F12)+RO(F22),

b. S1’’=RO(F11)+RO(F22) and S2’’=RO(F12)+RO(F21).



The above RO(…) function means „*original or reversed*”.

Line Diagram of Transaction Exchange

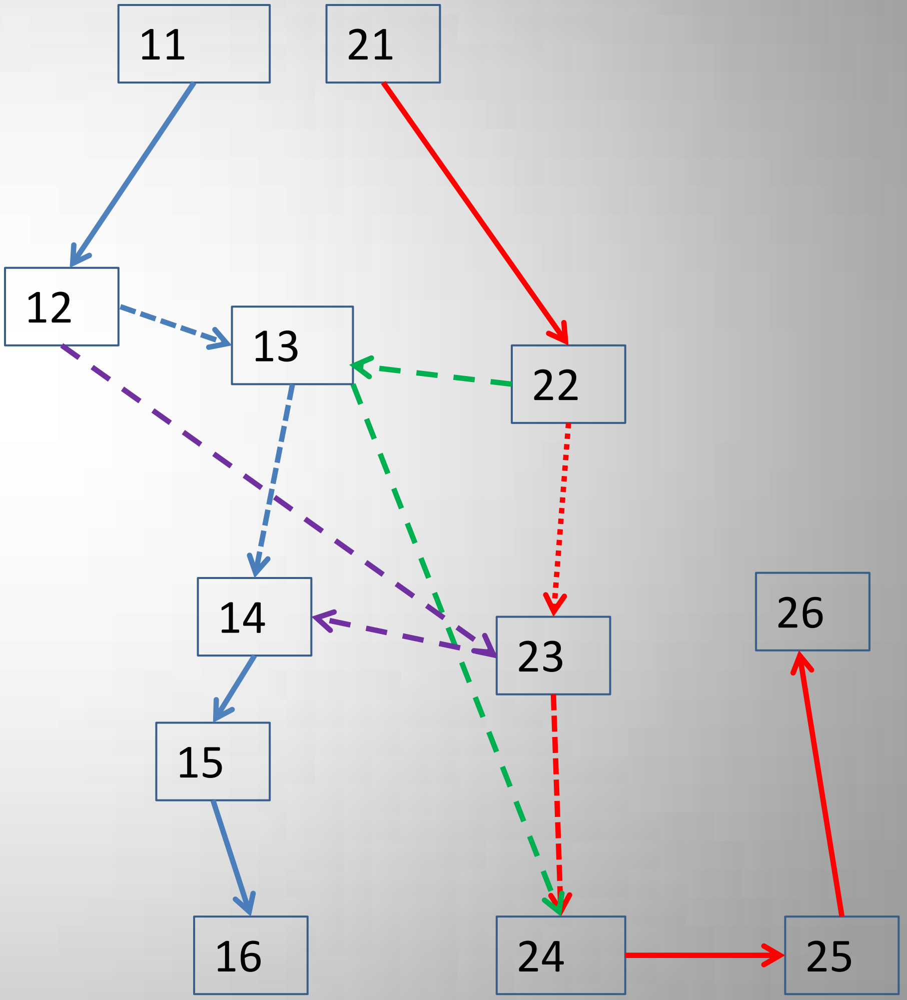
Exchange a place of an opened with a place of another opened route.

Routes before the transaction:

1) [11,12,13,14,15,16]
2) [21,22,23,24,25,26]

Routes after the transaction:

1) [11,12, 23, 14,15,16]
2) [21,22, 13,24,25,26]



Line Diagram of Transaction BRACE

1. Select two routes. They may be either open or closed ones.

2. Open them or cut them at an internal place.

3. Brace the routes at their both ends in order to get a single closed route.

Example

0. Two open routes before the transaction:

R1=[11,12,13,14,15,16]

R2=[21,22,23,24,25,26]

1. Cut edges: 14 → 15 and 24→25.

Fragments on the example:

F11 = [11,12,13,14] and F12 = [15,16]

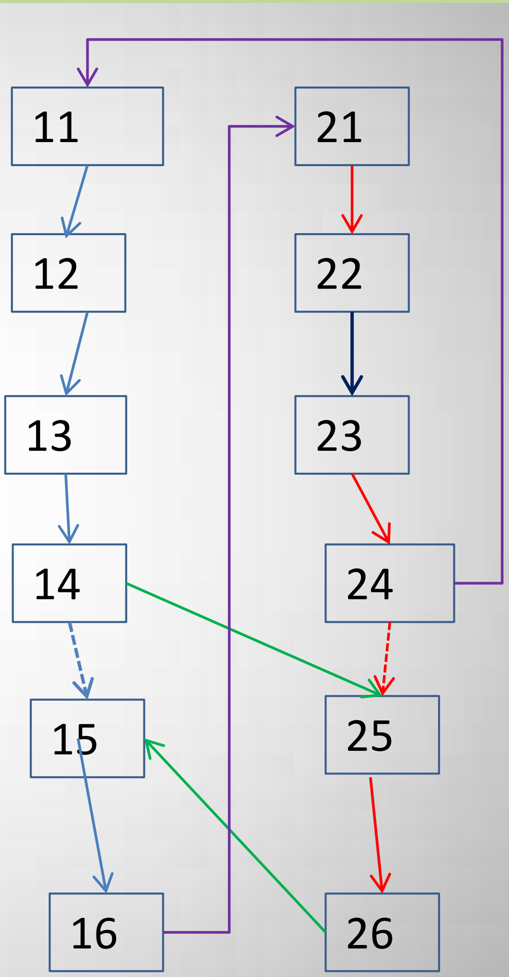
F21 = [21,22,23,24] and F22 = [25,26]

3. Bracing edges: 14 → 25, 26 → 15 ,

16 → 21, 24 → 11.

4. The result is the closed

[11,12,13,14, 25,26, 15,16, 21,22,23,24] route.



Line Diagram of the Transaction CLAMP

1. **Select two opened routes and an unused place.**

2. **Clamp the *head or the final* point of the first route *through the selected unused place* with the *head or the final point* of the second route.**

Example

0. Routes before the transaction:

R1=[11,12,13,14,15,16]

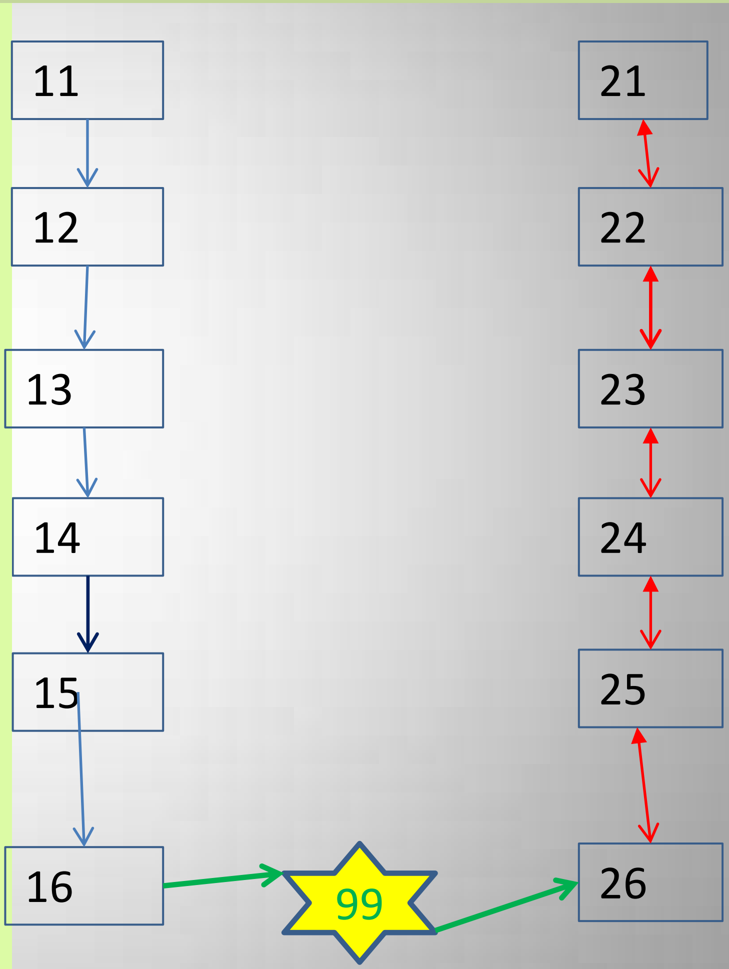
R2=[21,22,23,24,25,26]

1. Cut edges: none.

2. New edges: 16 → 99 and 99 → 26.

The clamped route is the opened

[11,12,13,14,15,16, 99, 26,25,24,23 22,21].



Undoing Transactions: Clip and Break

Transactions	State	Step	Process
Clip	Initial	0.	A closed route connecting at least three places.
	Intermediate	1.	Look for the endpoints of the longest connected[1] edge.
	Final	2.	An open route.
Break	Initial	0.	A closed route connecting at least seven places.
	Intermediate	1.	Look for the of shortest unconnected[1] place pair of the place set.
		2.	Look for the endpoints of the longest connected edge on the route.
		3.	Cut the found edges.
		4.	Cut the neighboring edges.
	Final	5.	Open route(s) and isolated place(s).

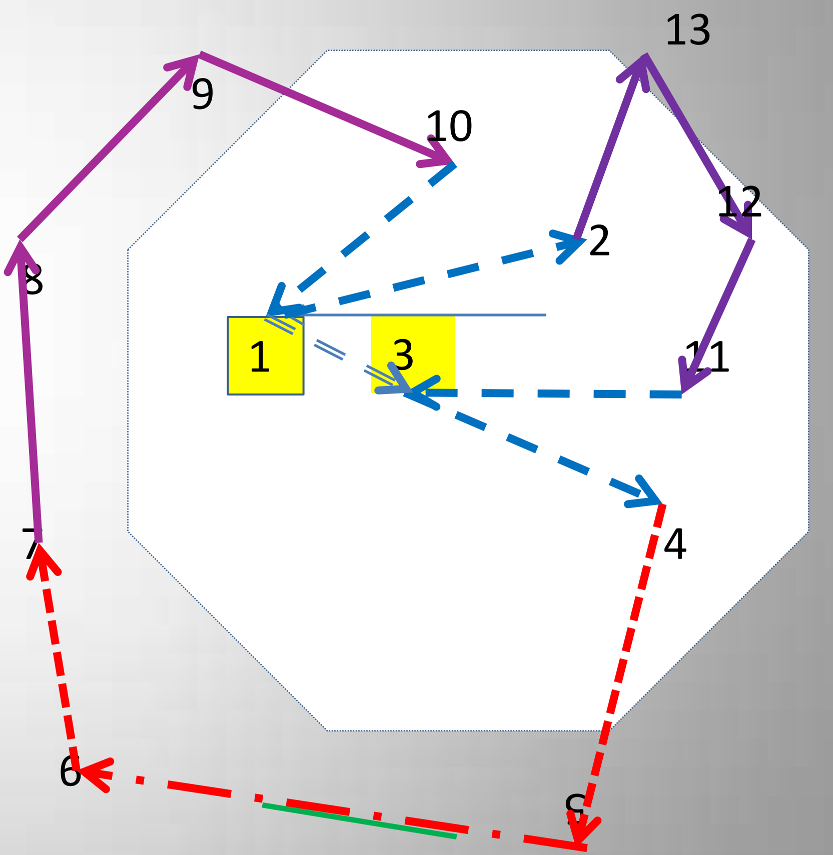
These transactions are not repeatable.

[1] They can use randomly corrected distances.

Line Diagrams of Transactions

BREAK and CLIP

Step	Routes before	Routes and isolated places after	Edges
B1.	***I:*** [1-2-13-12-11-3-4-5-6-7-8-9-10 | -1] ***closed.***		**Select** unconnected 1-3. ═ ═ ⇒
B2 and *C1.*			**Select** connected 5-6. ← — · —
B3.		1 and 3.	**Cut** 10-1,1-2, 11-3, 3-4. – – – – – → **Cut** 5-6, 4-5,6-7. ← — · — ← - - - - - -
B4.		***II:***[7 -8-9-10] ***open*** and ***III:*** [2-13-12-11] ***open.***	→→ ▬▬➡



The CLIP transaction cuts only the longest 5-6 edge.

Chapter 3

OUTPUT OF THE PROGRAM

Reported Tables

Saved Figures

Message Window

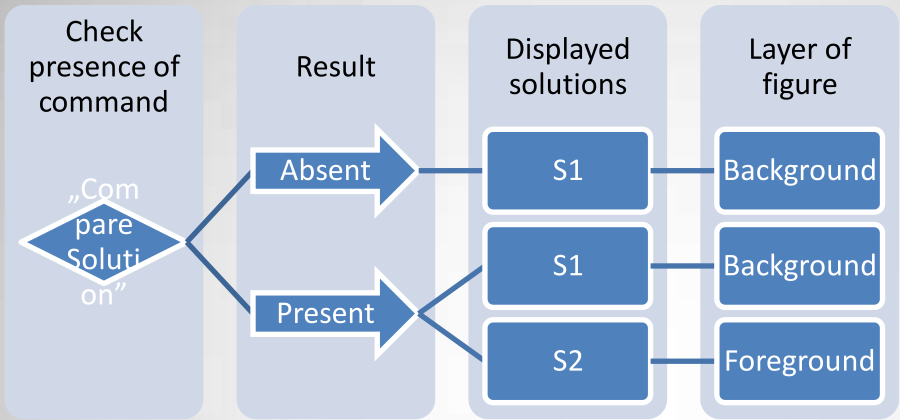
File Statistic and Status Line Toolbars

Next Chapter

Output Written into the Data Folder

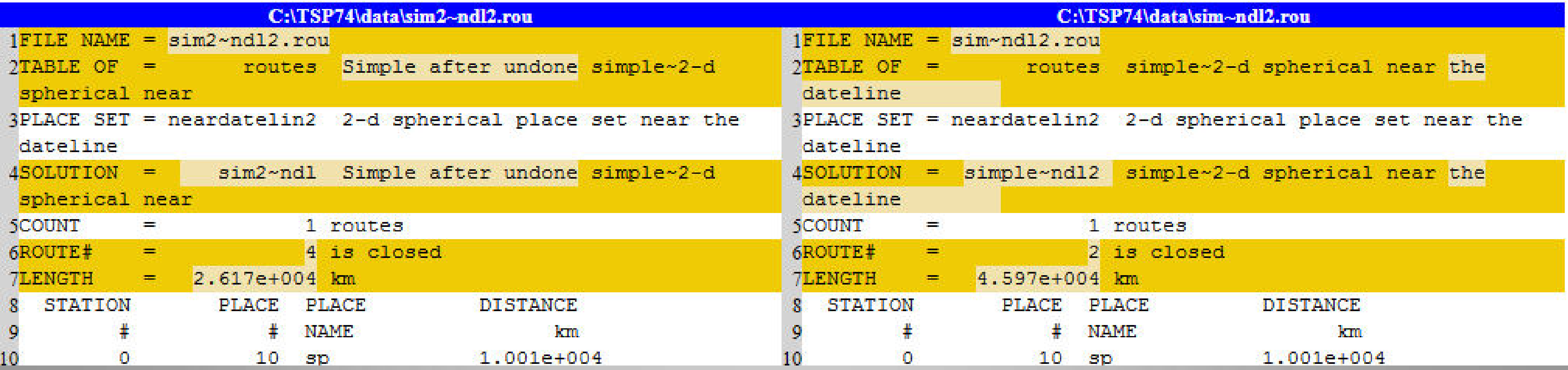
Root				Format	Extension	Contents
Start	Middle		End			
Fixed = Identifier of the solution	Time stampcontaining millseconds <yyMMddhhssqqq>	of creation		Text file	pla	Source PLACE SET.
					dis	Triangular distance matrix,
					for	Used plan and its transactions..
					rou	Final route(s) and their totall ength(s).
					tra	Allowed and executed number ofTRANSACTIONS.
					pro	Order of joining and cutting the edges.
					mrg	Two or more tables merged in a single file.
				Binary graphics	emf	FIGURE in extended metafile format. ***The user has to display, compress and convert it by his/her favorite tool.***
Fixed = „run”		of start of run		Prolog data base	fil	List of read and written input and output files. used_file(i,"connectonly1.tsr"). used_file(o,"run141127135917msg.html").
			Fixed = „msg”	Web page	html	Copy of the colored main message file*.* ***It may be a very big file Zip them to an archive or sellect all of their contents, copy to Excel andf save as binary files (*.xlsb).***
		of err		Text file	err	Call stack of the runtime errors.
Arbitrary user defined name				Batch input file	tsc, tsp, tsr	High level batch commands and batch commands generated by the interactive data entry dialogs; comments, rulers, error messages.

Layout of Comparative Reports and Figures



The lines of the layers are drawn with different colors.

Excerpt of a comparative report made by WinMerge and saved as HTML.



```
C:\TSP74\data\sim2~ndl2.rou
FILE NAME = sim2~ndl2.rou
TABLE OF  =        routes  Simple after undone simple~2-d
spherical near
PLACE SET = neardatelin2  2-d spherical place set near the
dateline
SOLUTION  =    sim2~ndl  Simple after undone simple~2-d
spherical near
COUNT     =           1 routes
ROUTE#    =           4 is closed
LENGTH    =   2.617e+004 km
  STATION       PLACE PLACE        DISTANCE
        #           # NAME               km
        0          10 sp           1.001e+004
```

```
C:\TSP74\data\sim~ndl2.rou
FILE NAME = sim~ndl2.rou
TABLE OF  =        routes  simple~2-d spherical near the
dateline
PLACE SET = neardatelin2  2-d spherical place set near the
dateline
SOLUTION  =  simple~ndl2  simple~2-d spherical near the
dateline
COUNT     =           1 routes
ROUTE#    =           2 is closed
LENGTH    =   4.597e+004 km
  STATION       PLACE PLACE        DISTANCE
        #           # NAME               km
        0          10 sp           1.001e+004
```

Chapter 4

USER INTERFACE

- Modes of user input
- Using batch commands collected in input files
- Batch command packets corresponding to each objects

Modes of User Input

The user can trigger the program

I. Using batch commands collected in input files described in the following pages.

II. Interactively

1. From the Task Menu of the program
2. Using the buttons of the Project Toolbar
3. Via the interactive dialogs.

This document deals only with the batch commands

The menus, buttons and dialogs are described in a separate document *tsp.chm on the author's homepage.*

Most of the batch commands have interactive counterparts.

Some interactive dialogs have not corresponding batch data packets.

Menu branch	**Log**	**Options**				
Menu item	All	Message Window	Input	Validation of Distances	Solution	Styles of Colored Reports
Batch packet	None	None	None	SOLUTION PLACE SET	None	None

Structure of Input Files

The input files consist of data packet started with the command line beginning with the verbs listed in the page of Summary of Command Order and closed by the corresponding *finish* line.

I. The data packets may contain

- The packets shall contain comment lines belonging to the currently handled object. *Their presence is recommended for better human readability of input data.*
- ruler lines showing the column names and boundaries of the subsequent command lines
- specific command lines whose *accepted verb+object pairs are described at the individual packets and tabulated together.*

II. Comment and ruler lines are allowed between packets, too.

III. Input redirection commands can be placed between packets and within certain packets. The input continues from the ceased input file when the end of included file is reached. They serve for

- including whole data packets
  - assembled by the user or
  - generated by a previous run of the TSP program, respectively.

Structure of Command Lines

I. **The information is arranged in the command records as a given count of fields of *n* × 12 characters. Their contents is trimmed and transformed to lowercase before evaluation. The capital letters of the *label* fields are retained only.**

II. The end of the command lines after the specific number of fields are ignored.

III. The instruction lines look like imperative sentences

1. with **a** **predicate** (verb) in their first field and (*n*=1)
2. with **an** **object name** in their second field (*n*=1).

IV. The third and subsequent fields may be

1. single

a) ***symbolic data*** with special set of legal contents or (*n*=1)

b) Unicode **string** data of restricted length or (*n*=1)

c) integer or real **numbers** in a special valid range (*n*=1).

2. merged

a) *labels* containing information about the objects (*n*=4)

b) ***arithmetic formulae*** consisting of variables constants and operators (*n*=3)

c) Unicode ***file names*** (*n*=3).

Kinds of Comment Lines

First non-blank character	Closing character (optional)	Embedded contents	Created in the	Displayed in the message window	Saved to the validated input
„{”	„}”	User’s comments*.	Previous run	Yes	Yes
			Current run	Yes	Yes
„<”	„>”	Rule names among delimiters.	Previous run	No	No
			Current run	Yes	Yes
„¿”	„?”	Error messages.	Previous run	No	No
			Current run	Yes	Yes

*The rest of the line until the total length of most complicated command is treated as comment that is until the 72nd character.

Input Redirection Command

Predicate	Object	Purpose	Nr. of fields	Length	Field name
INCLUDE	INPUT_FILE	Redirect command input to a given file.	1	36 chars	INPUT

Components of the file name

Base folder	*<Progdir>\..\Data*	If the program directory is *C:\TSP75\Exe* then the base folder is *C:\TSP75\Data.*
Default extension	*„tsr”*	„Ready files”
Other accepted extensions	*„tsp”* and *„tsc”.*	„source files” and „collected commands”
Unicity	The full file name must be different from the names of the current and higher included files.	

This command is accepted only between packets.
It is rejected within the packet starting commands and FINISH of any packet.

<table>
<tr><th>Contents</th><th>Type</th><th>Restrictions</th></tr>
<tr><td rowspan="2">Name and optional extension of an existing file name relative to the base folder.</td><td rowspan="2">Unicode string</td><td>Prohibited characters: ['/','>', '|', '<', ',', ':', '%','?','*'].</td></tr>
<tr><td>Prohibited character pairs:
doubled separators „\\”,”::”; space before a „.” or „\”.</td></tr>
</table>

Identification of PLACE SET Objects

Predicate	Object	Purpose	Nr. data fields	Validation rules of fields
MAKE	PLACESET	Make new set of places from scratch or based on an existing place set.	4	NEW_TITLE
				SPACE
				SYSTEM
				INI_TITLE
CONVERT	PLACESET	Convert the coordinate system of an existing place set and store the result in a new place set.	4	NEW_TITLE
				SPACE
				SYSTEM
				INI_TITLE
EDIT	PLACESET	Edit a variable set of places.	1	VAR_TITLE

Common Data of PLACE SET Objects

Predicate	Object	Purpose	Nr. of fields	Length	Validation rules
LABEL	PLACESET	Mark the place set with a long label.	1	48	LABEL
FILL	COORDINATE	Fill the given column of coordinates with a given common value in the new place set.	2	12	FiLLED_AXIS
				12	FILLED_VALUE
OMIT	COORDINATE	Omit the given column of coordinates from the new data.	1	12	OMITTED_AXIS
CALCULATE	DISTANCES	**Calculate distances using the chosen method.**	**1 or 2**	**12**	METHOD
		Correct great –circle distances by radial ones for 3d spherical place sets.			RADIAL_METHOD
VALIDATE	DISTANCES	Validate distances of the places in the evaluated set.	3	12	MIN_NEAREST
				12	MAX_FARTHEST
				12	RATIO_OVER

Addition of PLACE Object to Place Sets Using Cartesian Coordinates

Predicate	Object	Purpose	Number of dimensions	Length (chars).	Nr. of data fields	Validation rules of fields
ADD	PLACE	Add a place to the active place set using Cartesian coordinates.	Two	12	3	NEW_NAME
						X
						Y
			Three	12	4	NEW_NAME
						X
						Y
						Z

Addition of PLACE Object to Place Sets Using Spherical Coordinates

Predicate	Object	Purpose	Number of dimensions	Length (chars).	Nr. of data fields	Validation rules of fields
ADD	PLACE	Add a place to the active place set using Spherical coordinates.	Two	12	3	NEW_NAME
						LATITUDE
						LONGITUDE
			Three	12	4	NEW_NAME
						LATITUDE
						LONGITUDE
						RADIAL

Replacement of PLACE Object to Place Sets Using Cartesian Coordinates

Predicate	Object	Purpose	Number of dimensions	Length (chars).	Nr. of data fields	Validation rules of fields
REPLACE	PLACE	Replace a place in the active place set using Cartesian coordinates.	Two	12	3	OLD_NAME
						LATITUDE
						LONGITUDE
			Three	12	4	OLD_NAME
						X
						Y
						Z

Replacement of PLACE Object to Place Sets Using Spherical Coordinates

Predicate	Object	Purpose	Number of dimensions	Len. (chr).	Nr. of data fields	Validation rules of fields
REPLACE	PLACE	Replace a place in the active place set using spherical coordinates.	Two	12	3	OLD_NAME
						LATITUDE
						LONGITUDE
			Three	12	4	OLD_NAME
						LATITUDE
						LONGITUDE
						RADIAL

Deletion of PLACE Objects from PLACE SETS

Predicate	Object	Purpose	Number of data fields	Validation rules of fields
DELETE	PLACE	Delete a place from the active place set	1	OLD_NAME

End of Data for PLACE SETS

Predicate	Object	Purpose	Number of data fields
FINISH	PLACESET	End of this kind of data.	0

Order and Count of Commands for PLACE SETS

A) Make a New Set

Or-der	Predicate	Ex-clude each other	Condition	Need	Has defaults	Count	More than one same
#1	**MAKE**			Obligatory	No	= 1	Illegal
#2	**LABEL**			Obligatory	No	=1	Illegal
#3	**FILL**		***The new set has 2D spherical coordinates.***	Optional.	Yes	≤1	Illegal
			Otherwise	Neglected	---	*=0*	Illegal
#4	**CALCULATE**			Optional.	Yes	≤1	Illegal
#5	**VALIDATE**			Optional.	Yes	≤1	Illegal
#6	***ADD*** ***REPLACE*** ***DELETE***	*No*		*Optional*	*No*	*≥0*	*Mean new data*
#7	**FINISH**			Obligatory	---	=1	Illegal

Order and Count of Commands for Place Sets

B) Convert an Existing Set

Order	Predicate	Exclude each other	Need	Condition	Has defaults	Count	More than one
#1	CONVERT		Obligatory		No	=1	Illegal
#2	LABEL		Obligatory		No	=1	Illegal
#3	FILL OMIT	**Yes**	**Conditional**	Used in some conversions defined later.	**Yes**	**≤1**	**Illegal**
#4	**CALCULATE**		**Optional**		**Yes**	**≤1**	**Illegal**
#5	**VALIDATE**		**Optional**		**Yes**	**≤1**	**Illegal**
#6	FINISH		Obligatory		**---**	**=1**	**Illegal**

Order and Count of Commands for Place Sets C) Edit an Existing Set

Or-der	Predicate	Ex-clude each other	Need	Condition		Has defaults	Count	More than one same
				Ini tial count of places	Dimensions and coo. sys-tem			
#1	EDIT		Obligatory			No	= 1	Illegal
#2	LABEL		Optional			No	≤1	Illegal
#3	FILL		Condi-tional	*0*	*2D sph.*	Yes	= 1	Illegal
				Otherwise		---	=0	Neg-lected
#4	CALCULATE		Optional.			Yes	≤1	Illegal
#5	VALIDATE		Optional.			Yes	≤1	Illegal
#6	*ADD* *REPLACE* *DELETE*	*No*	Optional			*No*	*≥0*	*Legal*
#7	FINISH		Obligatory			---	=1	Illegal

Validation Rules for the Input Fields of Place Sets

Group	Type	Rule	Contents	Value set
Identifiers of the place sets	Unique Unicode strings	NEW_TITLE	Title of the new set of the places.	Nexisting place set identifier
		INI_TITLE	Places will be pasted from this earlier place set. This place set must contain at least one place.	Existing place set indentifier
		VAR_TITLE	Title of edited set.	
Identifiers of the individual places	Unique Unicode strings	NEW_NAME	*The first character of the name determines the subset within the whole place set in case of restricted solution of the problem.*	The name must be different from all earlier defined place names in the set.
		OLD_NAME	Any place identifier.	The name must be one of the earlier defined place names in the set.
		OLD_NAME1	Two different place identifiers.	
		OLD_NAME2		
Coordinates	Symbol	SPACE	Number of dimensions of made set.	{„two”, „three”}
		SYSTEM	Coordinate system of made set.	{„cartesian”, „spherical”}

Allowed Conversions of the Place Sets

Before conversion		After conversion							
Dim.	Sys.	Dim.	Sys.	Dim.	Sys.	Dim.	Sys.	Dim.	Sys.
		Two	Cartesian	Two	Spherical	Three	Cart.	Three	Sph.
Two	Cartesian	Rejected identity[1]		Impossible[6]		Allowed internal[3]		Impossible[6]	
	Spherical	Impossible[6]		Rejected identity[1]		Allowed internal		Allowed internal[5]	
Three	Cartesian	Allowed internal[2]		Possible only in two steps.[7]		Rejected identity[1]		Allowed internal	
	Spherical	Possible only in two steps. [8]		Allowed internal[4]		Allowed internal		Rejected identity	

See notes on the next slide.

Notes on the Conversions of PLACE SETS

[1] **The identity transformations are rejected within the CONVERT PLACESET packet. Use simple MAKE PLACESET data packet instead of CONVERT.**

[2] The omitted coordinate is defined by the OMIT COORDINATE command. The remaining two coordinates became X and Y coordinates.

[3] The name of new Cartesian coordinate and its common value are given by the „FILL COORDINATE <axis_name> <value>” command.

[4] The value of common spherical radius is given by the „FILL COORDINATE RADIUS <value>” command.

[5] The value of the common radius of the old place set is copied to the spherical radius coordinates of each new place.

[6] These conversions are impossible based on the original coordinates.

[7] P#1: convert the 3D Cartesian place set to 3D spherical; P#2: Convert the 3D spherical place set to 2D spherical .

[8] P#1: convert the 3D Spherical place set to 2D spherical; P#2: Convert the 2D spherical place set to 2D Cartesian

Usage of the „FILL COORDINATE" and „OMIT COORDINATE" Commands

Starting verb	Cnt . of pla-ces	Dimensions and coordinate system				„FILL COORDINATE" command				„OMIT COORDINATE"	
						Need	Fields			Need	Field
		Old		New			FILLED_AXIS	FILLED_VALUE			OMITTED_AXIS
							Value set	Range	Def.		V.set
MAKE	=0			2	Sph.	Optional	„radius"	$]-10^{-100},+10^{+100}[$	6371	Prohibited	---
				Any other		Prohibited	---	---		Prohibited	---
EDIT	=0	2	Sph.	2	Sph.	Optional	{„radius"}	$]-10^{+100},+10^{+100}[$		Prohibited	---
	>0	other		Unchanged		Prohibited	---	---		Prohibited	---
CONVERT	>0	2	Cart.	3	Cart.	Required	{„x", „y", „z"}	$]-10^{+100},+10^{+100}[$		Prohibited	---
		3	Cart.	2	Cart.	Prohibited	---	---		Required	{„x", „y" „z"}
		3	Sph.	2	Sph.	Required	{„radius"}	$]-10^{+100},+10^{+100}[$		Prohibited	---
		Any other				Prohibited	---	---		Prohibited	---
	=0	Any				Prohibited	---	---		Prohibited	---

Validation Rules for the LABEL Fields for All Objects

Rule	Contents	Type	Value set
LABEL	Label of the object	Non-empty string	The string is trimmed from both sides and its internal consecutive whitespaces are substituted with a single space. Its capitalization is not changed.

Validation Rules of the Symbols and the Values of the Coordinates

Name	Contents	Value set	
SPH_AXIS	Symbol of the filled common spherical coordinate value.	{„radius”}	
CAR_AXIS	Symbol of the filled or omitted axis of the Cartesian coordinate system.	{„x”, „y”, „z”}	
X	Cartesian coordinate X.	$]-10^{+100},+10^{+100}[$	
Y	Cartesian coordinate Y.		
Z	Cartesian coordinate Z.		
LATITUDE	Geographical latitude φ in degrees.	[-90.0,+90.0]	
LONGITUDE	Geographical longitude λ in degrees.	[-180.0,+180.0]	
RADIAL	Radial distance from the center of the sphere [km].	For the individual place coordinates	$]\ 0,+10^{+100}\ [$
		For the common radius	$[+10^{-100},+10^{+100}[$

Commands Identifying the PLAN Objects

Predicate	Object	Purpose	Demand	Nr. of data fields	Validation rules of fields
MAKE	PLAN	Make new plan	Required	3	NEW_PLAN
					FOR_SEARCH
					INI_PLAN
EDIT	PLAN	Edit a variable plan.	Required	2	VAR_PLAN
					FOR_SEARCH

Commands Handling the PLAN Objects

Predicate	Object	Purpose	Nr. of flds	Len. (chr)	Validation rules of fields
LABEL	*PLAN*	*Mark the plan with a long label.*	*1*	*48*	*LABEL*
ALLOW	TRANSACTION	Define repeatable transaction and number of its repetition within the search	2	12	REP_TRANSACT
				36	FORMULA
EXECUTE	TRANSACTION	Define non-repeatable transaction	1	12	NRE_TRANSACT
FINISH	PLAN	End of data for plan.	0		

Order and Count of Commands for PLANS

Order	Predicate		Exclude each other	Condition	Count	Second and further occurrences
#1	MAKE	EDIT	Yes		= 1	Illegal
#2	LABEL		Conditional	After MAKE	=1	Illegal
				After EDIT	≤1	
#3	ALLOW		No		≥ 0	The transaction symbols must be different.
#4	EXECUTE		No		≤ 2	
#5	FINISH		No		=1	Illegal

Validation Rules in packet PLAN

I. Identification of Plans and Transactions

Name	Contents	Type	Value set
NEW_PLAN	Identifier of the created plan of transactions.	Unique Unicode string	Different from earlier plan identifiers.
VAR_PLAN	Identifier of the updated plan of transactions.		Identifier of an earlier plan not referred in a done solution.
INI_PLAN	Transactions will be pasted from this earlier place set.		The reserved word *„empty”* or the identifier of an earlier defined plan.
FOR_SEARCH	Method of search	symbol	*{„full”, „greedy”, „undo”}*
NRE_TRANSACT	Non-repeatable transaction.	symbol	*Subset of {„permute”, „close”, „brace” , „clip”, „break”}* depending on the method of the search.
REP_TRANSACT	Repeatable Possible transaction of the above selected method.	symbol	All other transaction symbols.
FORMULA	Arithmetic expression of the maximal repeat count of the transaction.	Sequence of tokens	See in slide „Repeat Counts of Transactions”.

Symbols of Simple TRANSACTIONS in packet PLAN

The FORMULA field after the symbols contains a valid arithmetic expression detailed the count of repetition of the transactions.

Search	Symbol	Essence	Has formula field?
FULL	PERMUTE	Execute a full permutation of places.	*No, because it is not repeatable.*
UNDO	CLIP	Clip a closed route at its longest edge.	*No, because they are not repeatable.*
	BREAK	Break a closed route at more places.	
GREEDY	START	Start a new route from two free places.	Yes, because they are repeatable . The formula field contains a valid arithmetic expression.
	CONTINUE	Continue a route with a free place.	
	CONNECT	Connect two routes at their head or final points.	
	CLOSE	Close the remaining single route when no unused places are present.	*No, because it is not repeatable.*

The sophisticated transactions are treated in the next slide.

Symbols of Sophisticated TRANSACTIONS in packet PLAN

The repeatable sophisticated transactions need the FORMULA field after the symbols. This field contains a valid arithmetic expression detailed the count of repetition of the transactions.

Search	Repeatable	Symbol	Essence
GREEDY	YES	INSERT	Insert a single place between two places of the route.
		MERGE	Merge a whole route between two places of another route.
		EXCHANGE	Cut a place from both selected open route and insert the cut places at the original position of the other cut place.
		BRACE	Brace the routes at their both ends.
		SWAP	Join the fragments cut from different input in the best order.
		REVERSE	Cut a part of the closed route then insert in reversed order.
	NO	CLAMP	Clamp the two input routes across the selected place.

Validation Rules in packet PLAN
II. Repeat Counts of Transactions

*The **FORMULA** field must contain a valid arithmetic expression consisting of the following tokens.*

Tokens	Contents	Type	Value set
CONSTANT	Sequence of decimal digits	token	**N**
VARIABLE	The count of places of the set	symbol	{„p”}
OPERATOR	Infix, having two arguments	token or character	**{„+”, „-”, „*”, „^”, „mod”, „div”}**
	Unary		**{„+”, „-”}**
FUNCTION	**N** ↦**N** with one argument (result rounded down)	token	**{„sqrt”, „lg” }**
PARENTHESIS	Opening and closing parentheses	symbol	**{„(”, „)” }**

Examples of Arithmetic Expressions in Allowed Number of Transactions

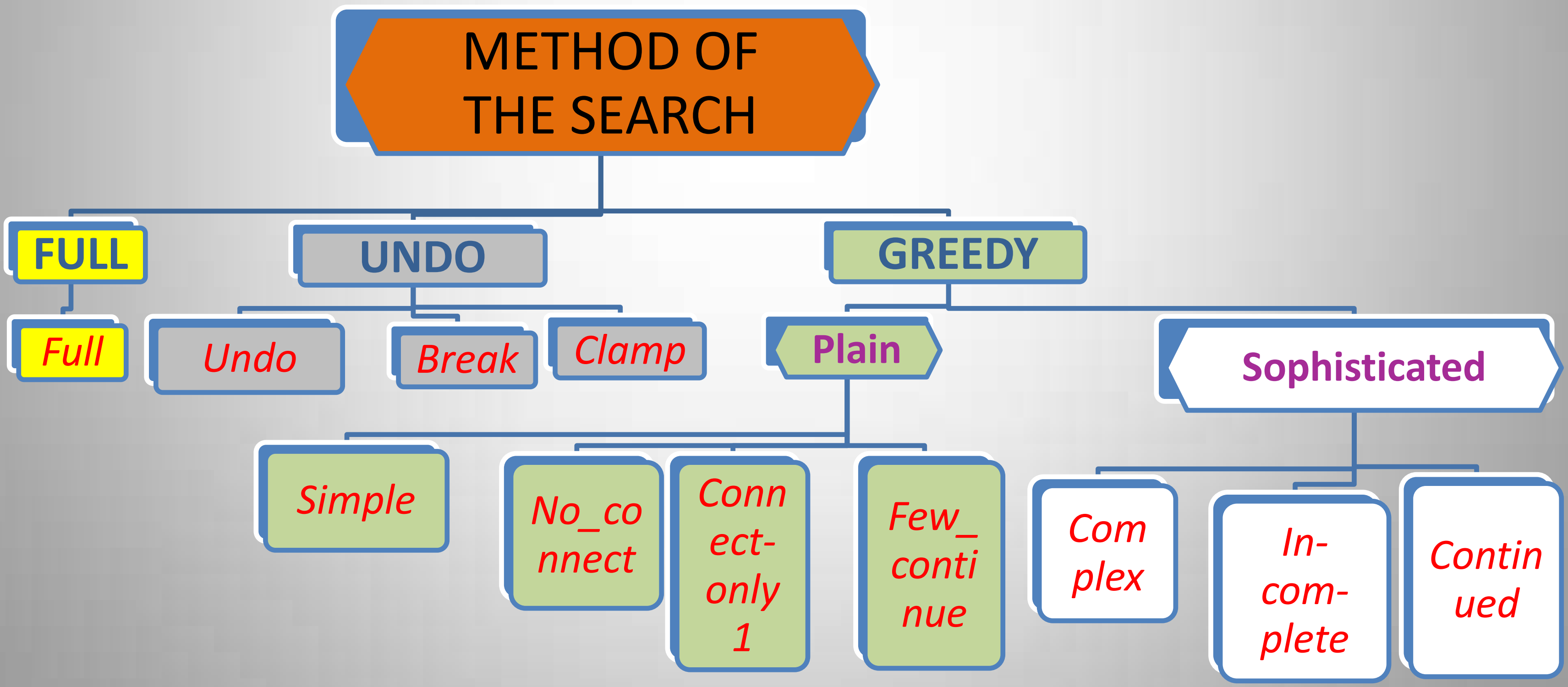
Valid

1. 0
2. 10
3. p
4. p – 3
5. p div 4 + 3
6. p mod 2 +p div 2 –1
7. lg p
8. p^2
9. sqrt (p div 2)
10. (p – 1) div 4

Invalid		Why?
i.	0.0	decimal dot
ii.	-	lonely operator
iii.	q	unknown variable
iv.	p-3-	unfinished expression
v.	p/3 – 5	invalid operator
vi.	p%3 – 5	invalid operator
vii.	ln p	illegal function
viii.	p**3	adjacent operators
ix.	sin p	illegal function
x.	(p-1))	badly nested parentheses

Ready Plans

The most frequently used plans can be read from ready input files. These plans are listed here and grouped **as shown in the second and third** line of the chart. They have to be read in from the corresponding *tsr* files by an **INCLUDE INPUT_FILE** command of the *data* folder before they can be referred in the *INI_PLAN* field.



Commands of SOLUTION Objects

Predicate	Object	Purpose	Flds	Lengths	Validation rules	
MAKE	SOLUTION	Define a solution starting from unconnected places.	3	12	OLD_TITLE	
					NEW_SOLUTION	
					OLD_PLAN	
LABEL	SOLUTION	Mark the solution with a long label.	1	48	LABEL	
CONTINUE	SOLUTION	Use calculated distances and intermediate results of an old solution.	1	12	OLD_SOLUTION	
CALCULATE	DISTANCES	Calculate distances using the chosen method.	1 or 2	12	METHOD	
		Correct great –circle distances by radial ones for 3d spherical place sets.		12	RADIAL_METHOD	
VALIDATE	DISTANCES	Validate distances of the places in the evaluated set.	3	12	MIN_NEAREST	
				12	MAX_FARTHEST	
				12	RATIO_OVER	
RESTRICT	SOLUTION	Restrict the solution to a subset of places having a given maximal distance of a central place.	3	12	FIRSTLETTERS	
BEGIN	PLACE	Begin the solution at the given place.	1	12	OLD_NAME	
FIX	EDGE	Connect the given places before the execution of the referenced plan. Do not allow to cut the given edge.	2	12	OLD_NAME1	Different place names are required.
				12	OLD_NAME2	
PROHIBIT	EDGE	Cut given places before the execution of the referenced plan. Do not allow to join the given edge.	2	12	OLD_NAME1	
				12	OLD_NAME2	
EXECUTE	PLAN	Execute the referenced plan in deterministic mode.	0			
RANDOMIZE	PLAN	Execute the referenced plan with randomized mode with thegiven ω smashing parameter.	1	12	SMASH_WIDTH	
FINISH	SOLUTION	Finish data packet, execute solution.	0			

Order and Count of Commands for SOLUTIONS

Order	Predicate		Exclude	Need	For search	Count	Repetition
#1	MAKE			Obligatory	All	=1	Illegal
#2	LABEL			Obligatory	All	=1	Illegal
#3	CALCULATE		Yes	Obligatory	All	=1	Illegal
#4	VALIDATE			Optional [1]	All	≤1	Illegal
#5		CONTINUE		Obligatory	All	=1	Illegal
#6	RESTRICT			Optional	All	≤1	Illegal
#7 In any order	BEGIN		No	Conditional	„GREEDY” and „UNDO” [2 3]	≤1	Illegal
	FIX		No			≥0	Allowed
	PROHIBIT		No			≥0	Allowed
#8	EXECUTE			Obligatory	All		Illegal
#9	FINISH			Obligatory	All	=1	Illegal

[1] The interactive counterpart of the CALCULATE+ VALIDATE commands is the Distance Calculation and Validation Options dialog.

[2] Each place name pair must be referred in only one FIX or PROHIBIT command

[3] The named places must not be out of the restricted area.

[4] The RANDOMIZE command changes a deterministic GREEDY or UNDO search to a randomized one. *It is illegal for **FULL** searches.*

Validation Rules of Identifiers within the SOLUTION Packet

Name	Contents	Type	Value set
OLD_PLAN	Identifier of a preset plan of transactions.	Unique Unicode string	Identifiers of the stored plans.
OLD_TITLE	Identifier of the solved place set.		Titles of the earlier defined place sets.
NEW_SOLUTION	Identifier of the new solution.		Unoccupied solution identifier.
OLD_SOLUTION	Identifier of the edited solution.		Identifier of an earlier solution **of the same data set.**
FIRSTLETTERS	Identifier of the central place of the restricted place set.	Unicode string	First letters of the place names included in the **restricted solution**.

Validation Rules of Distances - Solution Packet

Name	Contents		Type	Default value substituted in blank field		Value set
				Coordinate system		
				„cartesian”	„spherical”	
METHOD	Method of the distance calculation.		symbol	„eucledian”	„great_circle”	{„eucledian”, „manhattan”, „great_circle”}
RADIAL_METHOD	Correction of great-circle distance with the radius.		symbol	---	„eucledian”	{„eucledian”, „manhattan”}
SMASH_WIDTH	Smashing parameter ω of the randomized search.		Unsigned real	2.0		0.001<=SMASH_WIDTH<=10.0.
MIN_NEAREST	Bounds of distances from other places [km].	Lower bound for nearest.	Unsigned real	1 km		1<= MIN_NEAREST
MAX_FARTHEST		Upper bound for farthest.		5000 km	1.1*π*RADIUS_OF_EARTH==22017 km	2*MIN_NEAREST <= MAX_FARTHEST
RATIO_OVER	The place is invalid if the majority of other ones are too far from it.			0.75		0.5 <=RATIO_OVER<=1.0 See Validation of Places

Identification and Contents of a Simple Report

Predicate	Object	Purpose	Fld. cnt.	Len.	Fields
MAKE	REPORT	Report a given solution made of a given place set to the selected device split to separate tables or merged in a common file or windows, respectively.	4	12	EVA_TITLE
				12	EVA_SOLUTION
				12	DEVICE
				12	SPLIT
PRINT	TABLE	Select reported table.	1	12	TABLE
FINISH	REPORT	Finish REPORT packet.			

The reports inherit their labels from the included solutions.

The report files are written to the *<Progdir>\..\Data* folder. The roots of the automatically generated names are the concatenation of the identifier of the reported solution and the 15-digit timestamp of format *<yyyyMMddhhmmqqq>* . The extension of the report files is formed from the table names.

Comparison of Two Saved REPORT Files

The comparison is made externally using the WinMerge program.

Predicate	Object	Purpose	Fld. cnt.	Len*	Valida-tion rules of fields	Def. ext.	Base folder
INSPECT	REPORT	Name of first	1	36	INPUT	rou	*<Progdir>\..\Data* ------------------------ If the program directory is *C:\TSP75\Exe* then the base folder is *C:\TSP75\Data.*
COMPARE	REPORT	second compared report or data file	1	36	INPUT	rou	
FINISH	REPORT	Finish report packet.	0				

* The total length of reduced file name including the „.” and added default extension must not exceed 36 characters.

Order and Count of Commands for Simple Reports

Order	Predicate	Need	Count	Second and further occurrences
#1	MAKE	Obligatory	= 1	Illegal
#2	PRINT	Obligatory	≥ 1	Makes other table
#3	FINISH	Obligatory	= 1	Illegal

Order and Count of Commands for Comparative Reports

Order	Predicate	Need	Count	Second and further occurrences
#1	INSPECT	Obligatory	= 1	Illegal
#2	COMPARE	Obligatory	= 1	Illegal
#3	FINISH	Obligatory	= 1	Illegal

Validation Rules of Input Fields for Report and Figure Objects

Field name	Contents	Length, chars	Type	Value set
EVA_SOLUTION	Identifier of the (first) reported solution	12	Unique Unicode string	Identifier of an executed solution.
EVA_TITLE	Identifier of the (first) evaluated place set	12		Title of an earlier defined place set treated in the above solution.
SPLIT	Are the Report tables split?	12	symbol	{no: „merged”, yes: „separated”}
INPUT	Reduced name* of the inspected or compared existing Report or *any other text file.*	36	Unicode string	Prohibited characters before the dot of the extension: ['/', '>', '|', '<', ',', ':', "%”]
DEVICE	Output device	12	symbol	{„screen”, „disk”} **

* The base directory of the reduced file names is the data subfolder of the program directory that is the<*Progdir*>\..*Data* folder. **The extension of the output file is optional.** The root of the output file names i.e. the name without path and extension must be unique in the actual run of the TSP program.

** The Figures are written only to DISK as extended metafiles therefore only Reports need DEVICE. In the latest version of the program.

Possible Values of the Table Field of the Report Command

Symbol	Extension of the table	Reported table
„places”	„pla”	Source Place set.
„distances”	„dis”	Triangular Distance matrix calculated by the preset method.
„formulae”	„for”	Identifier of Plan and formulae for allowed transactions as functions of the *p* counter of places
„routes”	„rou”	Final route(s) and their total length(s).
„transactions”	„tra”	Summary of allowed and executed number of Transactions.
„process”	„pro”	Order of joining and cutting the edges between pairs of places.
„merged”	„mrg”	Merged file containing two or more kinds of tables.

Identification of FIGURE

Predicate	Object	Purpose	Fld. Cnt.	Length	Validation rules of fields
MAKE	FIGURE	Start of data packet giving the solution displayed in the background.*	2	12	EVA_TITLE
				12	EVA_SOLUTION
LABEL	FIGURE	*Long label.*	*1*	*48*	*LABEL*
COMPARE	SOLUTION	Identifier of compared solution displayed in the foregound	2	12	EVA_TITLE
				12	EVA_SOLUTION

* Two fields are deleted from the MAKE FIGURE command in the latest version of the program and the INCLUDE SOLUTION command is merged with it.

1. The figures are named automatically based on the identifier of the background solution and the timestamp of their creation.
2. Their device may be only disk.

Lookout and Display of FIGURE

Predicate	Object	Purpose	Flds.	Len.	Validation rules of fields
USE	VIEW	Use predefined settings and views	2	12	VAL_VIEW_ID
				12	VAL_SET_ID
FINISH	FIGURE	Finish data packet.			

Validation Rules of Fields in Commands Displaying FIGURES

Type	Field name	**Inormation contained**	**Range**	**Unit**
string	VAL_VIEW_ID	Identifier of valid predefined view	Unicode string	
string	VAL_SET_ID	Identifier of valid loaded settings	Unicode string	

Order and Count of Commands for Figures

Order	Predicate	Need	Condition	Count	Second and further occurrences
#1	MAKE	Obligatory		= 1	Illegal
#2	LABEL	Obligatory		=1	Illegal
#3	COMPARE	Optional	A foreground solution drawn over it	≤ 1	Illegal
#4	USE	Obligatory		= 1	Illegal
#5	FINISH	Obligatory		= 1	Illegal

Identification of SETTINGS

Predicate	Object	Purpose	Fld. Cnt.	Len	Validation rules of fields	Default and accepted extensions
LOAD	SETTINGS	Identify settings	2	12	NEW_SET_ID	
				36	INDIRECT_FILE	„ind”.
						Base folder: *<Progdir>\..\Data*
LABEL	SETTINGS	*Long label*	*1*	*48*	*LABEL*	
FINISH	SETTINGS	Close packet	0			

Order and Count of Commands for SETTINGS

Order	Command is	Count	Predicate	Object	Second and further occurrences
#1	Obligatory	= 1	LOAD	SETTINGS	**Illegal**
#2	Obligatory	=1	LABEL	SETTINGS	**Illegal**
#3	Obligatory	= 1	FINISH	SETTINGS	**Illegal**

Validation Rules for Identifiers and File Names for the SETTINGS and STYLES Packets

Type	Contents	Name	Value set
string	Identifier of new loaded settings	NEW_SET_ID	Unique unicode string
string	Name of the loaded indirect file containing the names of the files of individual settings	INDIRECT_FILE	Excluded characters= ['/', '>', '|', '<', ',', ':', "%"]
string	Name of the loaded style definition data base.	STYLES_FILE	

Consultable Files Containing Prolog Terms

- The Prolog terms start with a lowercase word [=functor], followed by an opening parenthesis, quoted strings, numbers and/or embedded terms separated by commas, finished by a closing parenthesis. The lines are closed by and period.

  -
    ```
    compose(p("Background","Cut Edges","Lines","Style"),s(1,"Dashed","~ (Pen & Screen)")).
    ```
- The consultable files begin with the „clauses” word and contain comments, empty lines and terms.
- The terms must be declared in the reading program properly.
- The files may contain comments between ‘%’ character and newline character.

  -
    ```
    style_setting(les(2,"table header",rgb(0,0,0),rgb(0,255,255),fo("Lucida Console","Eastern European",monospace,true,false,false,10))).
    ```

  -
    ```
    % 2014. 11. 16.14:20 Bg and fg of table header modified.
    ```

  -
    ```
    % 2014. 11. 16.14:17 Title and footer styles changed.
    ```
- This program consults with
  a. indirect files made by the **TestDraw** program described in the next pages
  b. then by the files of the settings mentioned in them;
  c. style definition data bases made by
     i. the **Colorful Report program**
     ii. or the TSP program itself.

The Indirect Files

The indirect files contain the terms referring to the consultable files containing the settings.

These files are generated by the TestDraw program.

Sample of indirect file

```
clauses
saved_as("nul").
indirect("Style","C:\\TestDraw74\\Exe\\on grey plot.savs").
indirect("Composition","C:\\TestDraw74\\Exe\\on grey plot.savc").
indirect("Areas","C:\\TestDraw74\\Exe\\neither top nor right scales.sava").
indirect("Location","C:\\TestDraw74\\Exe\\1.savl").
indirect("Font","C:\\TestDraw74\\Exe\\1.savf").
```

The first line containing the single „clauses” word is obligatory.

The saved_as(…) term is not reliable. It must be defined in the program but its content is unused.

Saved Settings and Styles

Role	Saved terms	*Extension*
Indirect list of files referring to files of	File names	**.ind*
- Rectangular areas of the figure	Area	**.sava*
- Color compositions	Color	**.savc*
- Font definitions and sample texts	Font	**.savf*
- Locations of the message window	Location	**.savl*
- Pen styles	Pen style	**.savs*
- Styles of colored message texts	Style_setting	**.xmp and 5 digits*

The structures of the terms are described in
the help file of the TESTDRAW and the **Colorful Report program**,
respectively.

Identification of STYLES

Predicate	Object	Purpose	Fld. Cnt.	Length	Validation rules of fields	Default and accepted extensions
LOAD	STYLES	Load a database containing the styles (fonts and colors) of the colored messages.	1	36	STYLES_FILE	*„xmp”; „xmm” and „xpp”+5 digits*
						Base folder: *<Progdir>\..\Data*
RESET	STYLES	Reset the default styles .	0			
FINISH	STYLES	Close packet	0			

Order and Count of Commands for STYLES

Order	Command is	Count	Predicate		Exclude each other	Object	Second and further occurrences
#1	Obligatory	= 1	LOAD	RESET	Yes	STYLES	**Illegal**
#2	Obligatory	= 1	FINISH			STYLES	**Illegal**

Identification of a New VIEW

Predicate	**Object**	**Purpose**	**Fld. Cnt.**	**Len.**	**Validation rules of fields**
MAKE	VIEW	Create a new view and fix it number of dimensions and coordinate system.	3	12	NEW_VIEW_ID
				12	SPACE
				12	SYSTEM
LABEL	VIEW	*Long label.*	*1*	*48*	*LABEL*

Identification of an Edited VIEW

Predicate	**Object**	**Purpose**	**Fld. cnt.**	**Len.**	**Validation rules of fields**
EDIT	VIEW	Edit an existing but not yet referenced view.	1	12	VAR_VIEW_ID
LABEL	VIEW	*Long label.*	*1*	*48*	*LABEL*

The number of dimensions and the coordinate system cannot be corrected.

Setting the Shown Range of VIEWS

0. General Rules

- **This commands have three obligatory common symbolic fields**
  1. Verb
  2. Object
  3. Coordinate (axis)
- **and two conditional real fields**

  4. Constraint

  5. Other constraint.
- **The required number of commands setting the shown ranges is**
  - i. Is less than or equal to the number of dimensions of the view.
  - ii. At most one range setting command has to belong to each coordinates. If more range settigs are found for a coordinate the onyl the last one is valid.
  - iii. If the range setting is missing for a coordinate then the implicit „FIT" setting is used.

1. **The legal verbs in the first field are**
   - *A. „fit",*
   - *B. „limit",*
   - *C. „center".*
2. **The second fields has to contain *„view"* object name.**
3. **The third field contains the symbol of the axis whose accepted values depending on the**
   - *i. number of dimensions and*
   - *ii. the coordinate system*

   have been set in the *„MAKE VIEW"* command .
4. **The shown ranges of the coordinates are in fields #4 and #5.**
   - A. **These ranges are neglected and substituted with zeroes** after the **„FIT"** verb;
   - B. the lower limit for all coordinates but ***longitude"*** ,
   - C. the central meridian for the longitude coordinate.
5. **The other value of the range**
   - A. **These ranges are neglected and substituted with zeroes** after the **„FIT"** verb;
   - B. the upper limit for all coordinates except longitude",
   - C. the extent of the shown range of longitudes.

Setting the Shown Range of VIEWS

I. Two-dimensional PLACE SETS defined in Cartesian Coordinates

Predicate	Object	Purpose	Flds.	Len. chr.	Contents	Validation rules of fields
LIMIT	VIEW	Set lower and higher limits of the view.	3	12	Coordinate	COOR_TWO_CAR
				12	Lower limit	LOWEST_XYZ
				12	Upper limit	HIGHEST_XYZ
FIT	VIEW	Fit the range of the given coordinate to the displayed place sets when the figure is drawn.	1	12	Coordinate	COOR_TWO_CAR
CENTER	VIEW	Set center and extent of the view.	3	*12*	Coordinate	COOR_TWO_CAR
				12	Central meridian	CENTER_XYZ
				12	Extent of longitude	EXTENT_XYZ

Setting the Shown Range of VIEWS

II. Three-dimensional PLACE SETS defined in Cartesian Coordinates

Predicate	Object	Purpose	Nr. of flds.	Len. chr.	Contents	Validation rules of fields
LIMIT	VIEW	Set lower and higher limits of the view.	3	*12*	Coordinate	COOR_THREE_CAR
				12	Lower limit	LOWEST_XYZ
				12	Upper limit	HIGHEST_XYZ
CENTER	VIEW	Set center and extent of the view.	3	*12*	Coordinate	COOR_TWO_SPH
				12	Central meridian	CENTER_XYZ
				12	Extent of longitude	EXTENT_XYZ
FIT	VIEW	Fit the range of the given coordinate to the displayed place sets when the figure is drawn.	1	12	Coordinate	COOR_THREE_CAR

Setting the Shown Range of Views

III. Two-dimensional Place Sets defined in Spherical Coordinates

Predicate	**Object**	**Purpose**	**Nr. of flds.**	**Len.**	**Contents**	**Validation rules of fields**	
						„latituutude”	**„longitude”**
LIMIT	VIEW	Set lower and higher limits of the view.	3	*12*	Coordinate	COOR_TWO_SPH	
				12	Lower limit	LOWEST_LAT	LOWEST_LON
				12	Upper limit	HIGHEST_LAT	HIGHEST_LON
CENTER	VIEW	Set center and extent of the view.	3	*12*	Coordinate	COOR_TWO_SPH	
				12	Central meridian	CENTER_LAT	EXTENT_LAT
				12	Extent of longitude	EXTENT_LON	EXTENT_LON
FIT	VIEW	Fit the range of the given coordinate to the displayed place sets when the figure is drawn.	1	12	Coordinate	COOR_TWO_SPH	

Setting the Shown Range of VIEWS

IV. Three-dimensional PLACE SETS defined in Spherical Coordinates

Predicate	Object	Purpose	Nr. of flds.	Len.	Contents	Validation rules of fields		
						„latitude"	„longitude"	„radial"
LIMIT	VIEW	Set lower and higher limits of the view.	3	*12*	Coordinate	COOR_THREE_SPH The validation rules of the following fields depend on its value .		
				12	Lower limit	LOWEST_LAT	LOWEST_LON	LOWEST_RAD
				12	Upper limit	HIGHEST_LAT	HIGHEST_LAT	HIGHEST_RAD
CENTER	VIEW	Set center and extent of the view.	3	*12*	Coordinate	COOR_THREE_SPH		
				12	*12*	CENTER_LON	CENTER_LAT	CENTER_RAD
				12	*12*	EXTENT_LON	EXTENT_LON	EXTENT_RAD
FIT	VIEW	Fit the range of the given coordinate to the minimal and maximal values of the displayed	1	12	Coordinate	COOR_THREE_SPH		

Scaling, Mapping and Projection

Predicate	Object	Purpose	Flds.	Len.	Information in the field	Fields and Validation rules
OMIT	COORDINATE	Omit a coordinate from the three instead of projection.	1	12	Symbol of the omitted coordinate.	Depends on the coordinate system and number of dimensions
					Invalidate earlier given omission.	„none”
PROJECT	SPACE	Select collineation of three-dimensional data to a plane.	1	12	Explicit or implicit method of collineation.	COLLINEATION
MAP	SPHERE	Select map projection of the surface of the sphere.	2	12	Explicit or implicit mode of mapping.	MAPMODE
				12	Central pole of the mapping.	POLE

End of VIEW Packet

Predicate	Object	Purpose	Flds.
FINISH	VIEW	Close packet.	0

Order and Count of Commands within the VIEW Packets I.

Order		Predicate		Exclude each other	Need	Condition	Count	Second and further occurrences
Group	Within group						Total	
I. Identification	#1	MAKE	EDIT	Yes	Obligatory	Select one of them	= 1	Illegal
	#2	LABEL		No	Conditional	After MAKE	=1	Illegal
						After EDIT	≤1	
	Grand total within the group						≤2	
II. Shaping	#1	OMIT		No	Optional	See „Allowed Sequences of Projection Commands”.	≤1	Illegal
	#2	MAP		No	Optional		≤1	
	#3	PROJECT		No	Optional		≤1	
	Grand total within the group						≤3	

Order and Count of Commands within the VIEW Packets II.

Order		**Predicate**	**Exclude each other**	**Need**	**Condition**		**Count**		**Second and further occurrences for the same coordinate**
Group	Within group						For a given coordinate	Total	
III. Range of displayed places	#1	LIMIT	Yes if the axis name is the same.	Optional	Number of dimensions	=2	≤1	≤2	Overwrite first ones.
						=3	≤1	≤3	
						=2	≤1	≤2	
	#2	CENTER				=3	≤1	≤3	
	#3	FIT				=2	≤1	≤2	
						=3	≤1	≤3	
	Grand total within the group					=2	=1	=2	
						=3	=1	=3	
IV. Final	Last	FINISH	No	Obligatory				=1	Illegal

Allowed Sequences of Projection Commands

ABBREVIATIONS							
LINE		PROJECT ION		COORDINATE		MAPMODE	
Q	Question	i	Isometric	Lo	Longitude	c	cylindirical
A	Answer	c	Cavalier	La	Latitude	p	Postel
S	Sequence	p	Per Perpecti			s	sinusoidal
C	Command	t	Rot			x	xyz
O	Object	r	recommended			r	recommended

Q 1: Coordinate System

A 1: Cartesian | Spherical

Q 2: N. dim. | Number of dimensions

A 2: 2 | 3 | 2 | 3

S .: A | B | C | D | E | G | H

C 1: OMIT | PROJECT | MAP | OMIT | OMIT | PROJECT

O 1: X | Y | Z | i | c | p | t | r | c | p | s | r | Lo | La | Radial | Isometric | Cavalier | Perspectivic or Rotapers | Recom-mended

C 2: MAP | MAP | MAP | MAP | MAP

O 2: c | p | s | r | c | p | s | x | r | c | p | s | x | r | c | p | s | x | r | c | p | s | r

See methods of projection and mapping.

Accepted and Default Values of COLLINEATION Field

This fields contains the method of projection of the place set.

If this command is missing then the recommended of projection is used.

If recommended projection is not found then the default value is used.

The available projections are described in the chapter of algorithms.

Value type	**Number of dimensions and coordinate system**	
	3-dimensional spherical coordinates	**otherwise**
Recommended value	Depends on presence of extreme relative radial distances.	„isometric”
Default value	„cavalier”	„isometric”
Explicit values	*„isometric”* instead of „isometric axonometric”	
	„cavalier” instead of „modified Cavalier”	
	„perspectivic” instead of „perpendicular perspectivic”	
	„rotapers” instead of „rotated perspectivic”	

Accepted and Default Values of MAPMODE and POLE Fields

This fields contains the method of mapping of the spherical coordinates.

If this command is missing then the default methods of projection are used.

If recommended projection is not found then the default value is used.

The recommended mode is determined as described in the chapter of algorithms.

The POLE FIELD is optional. It is used only with the Postel projection.

Value type	**Value set of MAPMODE**	**Value set of POLE**	**Availability**
Recommended value	„recommended”		based on the distribution of latitudes.
Default value	„cylindrical”	„northern”	
Explicit values	„sinusoidal”	„northern”	
	„xyz”	„northern”	3-d spherical view without omitted axis.
	„postel”	„northern”	No places in the opposite hemisphere
		„southern”	
	„cylindrical”	„northern”	No places near the poles.

Validation Rules of VIEWS

I. Cartesian Coordinate System

1. Rules involving only one field

Name	*Contents*	*Default value*	*Type and unit*	*Value set*
LOWEST_XYZ	Lower limit of the view for the given coordinate	1.0	real, km	] −∞,+∞[
HIGHEST_XYZ	Higher limit of the view for the given coordinate	2.0		
CENTER_XYZ	Center of the view	1.0		
EXTENT_XYZ	Extent of he view	2.0	real, km	] 0,+∞[

2. Rules involving two fields

Only LOWEST_XYZ < HIGHEST_XYZ is accepted.

II. LIMIT Fields of the Spherical Coordinate System

1. Rules involving only one field

Type and unit	Coordinate	Quality	Name	Default value	Value set	Meaning
Real, degrees	Latitude	Lowest	LOWEST_LAT	1.0	[–90;+90]	Use these values as limits of shown latitudes.
		Highest	HIGHEST_LAT	2.0		
Real, degrees	Longitude	Lowest	LOWEST_LON	1.0	[–180;+180]	Use these values as limits of shown longitudes.
		Highest	HIGHEST_LON	2.0		
Real, km	Radial distance	Lowest	LOWEST_RAD	1.0	[0,+∞ [	Use these values as limits of show radial distances.
		Highest	HIGHEST_RAD	2.0	] 0,+∞ [	

2. Rules involving two fields

i. Only *LOWEST_LAT* < *HIGHEST_LAT* is valid.
ii. Only *LOWEST_LON* < *HIGHEST_LON* is valid.
iii. Only *LOWEST_RAD* < *HIGHEST_RAD* is valid.

Validation Rules for Identifiers for the VIEW packets

Type	Contents	Name	Value set
String	Identifier of newly defined view	NEW_VIEW_ID	Unique Unicode string
String	Identifier of defined view which has not been referenced in a figure	VAR_VIEW_ID	Unique Unicode string
Symbol	Number of dimensions	SPACE	*„two”* or *„three”*
Symbol	Coordinate system	SYSTEM	*„Cartesian”* or *„spherical”*

Accepted Values of COORDINATE NAMING Fields

These fields are present in the OMIT COORDINATE and the LIMIT VIEW , CENTER VIEW commands.

Coor. system	Cartesian		Spherical	
Dimensions	**2**	**3**	**2**	**3**
Field name Values	COOR_TWO_CAR	COOR_THREE_CAR	COOR_TWO_SPH	COOR_THREE_SPH
„x”	Valid	Valid	Invalid	Invalid
„y”	Valid	Valid	Invalid	Invalid
„z”	Invalid	Valid	Invalid	Invalid
„latitude”	Invalid	Invalid	Valid	Valid
„longitude”	Invalid	Invalid	Valid	Valid
„radial”	Invalid	Invalid	Invalid	Valid

Validation Rules of CENTERS and EXTENTS of VIEW

I. Spherical Coordinate System

Field(s)	***Contents***	***Type and unit***	***Value set***
CENTER_LAT	Center of the view for the latitudes	Real, degrees	[-90,+90]
EXTENT_LAT	Extension of the view for the latitudes.	Real, degrees	] 0,+90]
CENTER_LAT-EXTENT_LAT	Lower limit for the latitudes.	Real, degrees	[-90,+90[
CENTER_LAT+EXTENT_LAT	Upper limit for the latitudes.	Real, degrees	]-90,+90]
CENTER_LON	Central meridian of the view for the longitudes	Real, degrees	[-180,+180]
EXTENT_LON	Extension of the view for the longitudes.	Real, degrees	] 0,+180]
CENTER_RAD	Center of the view for the radial distances.	Real, km	] 0,+∞ [
EXTENT_RAD	Extension of the view for the radial disances.	Real, km	] 0,+∞ [
CENTER_RAD-EXTENT_RAD	Lower limit for theradial distances.	Real, km	[0,+∞ [

Table of Understood Verbs

The first 12 characters of the command lines may contain the following **verbs** after trimming and converting them to lower case:

Verb	Verb	Verb	New verbs
add	execute	make	begin
allow	fill	map	fix
calculate	finish	omit	prohibit
center	fit	print	restrict
compare	include	project	randomize
continue	inspect	replace	validate
convert	label	use	reset
delete	limit		
edit	load		

Table of Understood Objects

The second 12 characters of the command lines may contain the following **objects** after trimming and converting them to lower case :

Object	Object
coordinate	settings
distances	solution
figure	space
input_file	sphere
place	styles
placeSet	table
plan	transaction
report	view
edge	

General Sequence of Steps within the Data Packets

I. Each data packet has at most three two possible starting commands, typically *„make <object>”* and *„edit <object>”*. See table on the next page.

II. The packets making objects usually must be continued with a mandatory *„label <object> <non-empty text>”* command. See table on the next page.

III. The edited packets may be continued by an optional *„label <object>”* command.

IV. Some properties of the objects cannot be later changed therefore some verb+object combinations are Rejected identity in „edit <object>” commands.

V. Some properties of the objects are valid only in certain number of dimensions and/or coordinate system.

VI. Some commands may be mixed and/or repeated.

VII. There are commands in certain packets which can be followed exclusively by the „finish <object>” command.

VIII. Each packet has to be closed by the corresponding „finish <object>” command.

Summary of Command Order

The validity of the verb+object pairs of the command lines depend on the steps determined by the previous commands. The table of status changes is published as an Excel table.

Start and End of Packets and Labeling

Verb		Object name = Packet name:								
		figure	placeSet	plan	report	styles	settings	solution	view	Total
START	continue							LM2		1
	edit		LO2	LO2					LO2	3
	inspect				L--					1
	load					L-	LM2			2
	reset					L-				2
	make	LM2	LM2	LM2	LM2			LM2	LM2	6
	convert		LM2							1
END	**finish**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**9**

Legend	LM2	*The „label <object> <text>" is the mandatory second command of the packet.*
	LO2	*The „label <object> <text>" is an optional second command of the packet.*
	L--	*The packet does not contain the „label <object> <text>" command.*

Commands Allowed in Certain Cases

- Some packets contain such commands which can be issued only after starting by „make” verb, e. g.
  - *„calculate distances”,*
  - *„correct distances”*. See more at the description of the data packets.
- Some properties of the objects are valid only in certain number of dimensions and/or coordinate system, e. g.
  - *„insert coordinate”,*
  - *„omit coordinate”.*
  - *„calculate coordinate”;*
  - *„correct distances”,*
  - See more at the description of the data packets.