Cogn Process (2017) 18:145–157
DOI 10.1007/s10339-016-0788-z

RESEARCH REPORT

Cognitive dimensions of *talim*: evaluating weaving notation through cognitive dimensions (CDs) framework

Gagan Deep Kaur[1]

Received: 29 March 2016 / Accepted: 12 December 2016 / Published online: 28 December 2016

Abstract The design process in Kashmiri carpet weaving is distributed over a number of actors and artifacts and is mediated by a weaving notation called *talim*. The script encodes entire design in practice-specific symbols. This encoded script is decoded and interpreted via design-specific conventions by weavers to weave the design embedded in it. The cognitive properties of this notational system are described in the paper employing cognitive dimensions (CDs) framework of Green (People and computers, Cambridge University Press, Cambridge, 1989) and Blackwell et al. (Cognitive technology: instruments of mind—CT 2001, LNAI 2117, Springer, Berlin, 2001). After introduction to the practice, the design process is described in 'The design process' section which includes coding and decoding of talim. In 'Cognitive dimensions of talim' section, after briefly discussing CDs framework, the specific cognitive dimensions possessed by talim are described in detail.

Keywords Talim · Cognitive dimensions · Design process · Kashmiri carpet weaving · Carpet designing · Graph

Introduction

The design process in Kashmiri carpet weaving is mediated by a weaving notation called *talim* (pronounced *taa'leem*) which encodes the entire design in symbols. The Kashmiri carpet weaving is hand-knotted pile carpet weaving exhibiting curvilinear patterns in its designs. The practice traces its lineage to Persia from where skilled artisans are held to be brought down around sixteenth century by the Kashmiri ruler Zain-ul-Abidin (Mathur 2004: 18; Gans-Reudin 1984: 14, 31). However, talim is held to be of Kashmiri innovation (Harris 2001; Roy 2004: 225) where it was used earlier in shawl weaving and later, around eighteenth century, got adapted to carpet weaving (Saraf 1987: 89; Sajnani 2001: 161). Nowadays, it is also held to be used in Amritsar carpet weaving (Gans-Reudin 1984; Harris 2003) where it started due to migration of Kashmiri artisans in nineteenth century (Leitner 1882).

The hand-knotting technique necessitates the existence of a premeditated structure which may guide the weaver as to which colored knots should be woven and in how many numbers, so that an intricate design emerges as the weaving proceeds. This structure is the talim: it is systematic coding of design instructions which the weaver follows during weaving; rather, it is widely held among the community that no carpet can be woven without talim. The talim is, thus, unlike musical notations which can be dispensed within musical performances, at least, in Indian music traditions.

Methodology This study seeks to understand the nature of situated and distributed cognitive processes in Kashmiri carpet weaving with special attention to the negotiation of talim by different actors in their particular task domains. Hence, an in-depth understanding of the *design process*, during which talim is generated, and *weaving*, during

Handling Editor: Ekaterina Shutova (University of Cambridge).

Reviewers: Alan Blackwell (University of Cambridge), Marian Petre (The Open University).

Gagan Deep Kaur
gaganrism@gmail.com

[1] Consciousness Studies Programme, National Institute of Advanced Studies (NIAS), Bengaluru, India

which talim is decoded, is pertinent. For that, the methodology of cognitive ethnography is adopted and fieldwork to this effect has been conducted in 2015 (May–November) and in 2016 (April–November) in Srinagar, Kashmir, using methods like *participant observation* in which the researcher learned designing, coding and weaving from the expert respondents; *document analysis*; a mix of *semi- and unstructured interactions* with the community; *literature review*; and *videography* of constituent activities.[1]All interactions take place in Hindi in which both researcher and respondents are fluent and are audio/video recorded, wherever permitted. While a few respondents work in governmental setups, a majority of them work either in private establishments or as freelancers. The interactions center on cognitive, creative and situated aspects of the particular task domain of the respondent like designing, coding and weaving, and any other aspect the respondents might find interesting. At times, themes from cognitive dimensions framework used to be floated by the researcher to anchor the discussions around cognitive aspects. First, the design process describes the arrangement and roles of actors, their task contexts, the coding and decoding of talim:

The design process

The design process in Kashmiri carpet weaving is distributed over a number of actors and artifacts making it a *distributed design system* and is undertaken in two different settings: manual and computer assisted. Though the number of actors and artifacts differ considerably in both, a similarity is observed due to talim, which is identical in both settings, and is likewise processed by the weavers. I restrict here to manual setting as code generation is revealed in this setting only. This setting is prevalent traditionally and has little changed since its first recording by Moorcroft (1841: 188) and Leitner (1882: xxv)—both in the context of shawl weaving and later by Lawrence (1895: 377) in the context of carpet weaving. The similarity of their accounts shows identical nature of the design process in shawl and carpet weaving domains. Further, the talim samples given by Leitner (1882) for shawl (p.6) and carpet weaving (p.18) prove similar talims used in both which is followed till date. I restrict to carpet design process and the usage of talim in it. For talim's usage in shawl weaving, Harris (1991, 2000, 2001, 2007) is a good source.

The first actor in the design process is the designer (*naqash)* who creates pencil-drawn designs on graphs. The graphs are chosen as representational medium because their grid structure of horizontal–vertical lines parallels the warp-and-weft thread scheme on the loom.[2] The currently used graphs, i.e., the inch-square graphs, were devised in 1990s.[3] The evaluating criteria for a Kashmiri carpet are its knottage, i.e., number of knots per square inch (hereafter psi) and its size measured in feet. A knottage count of 16 × 16, i.e., 256 knots psi, to 30 × 30, i.e., 900 knots psi, is commercially available, with the most common knottage system being 20 × 20, i.e., 400 knots psi. In an inch-square graph, every inch square comprises of 16 sub-blocks of $5 \times 5 = 25$ cells which are evenly distributed with 4 sub-blocks in each row and column in the grid, giving us a total of 400 cells in the grid. This corresponds with our 20 × 20 knottage. Thus, these graphs are the most precise representation of the weave structure devised so far in the practice which can accommodate higher or lower knottages than 20 × 20 as well.

Thus, the task of a designer starts with *calculating the number of graphs* required for drawing the design of a particular knottage for a particular size. Once this is done, the designer creates pencil-drawn designs on these graphs according to specific *design types*. For instance, in a *kashyan* design type, only one quarter of the design is created and the remaining three quarters involve mirror images of that quarter which are woven by interpretation imposed by the weaver on the code (Fig. 1). Once drawing is done, the designer gives color scheme by assigning *color codes* on these pencil-drawn motifs. See the following design *portion,* its corresponding design and talim:

At this point arrives the second actor, namely the talim writer (*talim guru)* who writes, in specific symbols, these color codes plus the number of knots to be woven with these colors systematically on a long strip of paper—a process called *'talim uthana'* or 'picking the codes' from the graph. This coded script is called *talim* in which one unit of code comprises *number of knots plus color code*. One column in a talim comprises four blocks with twenty rows and represents one-inch square in the graph, if inch-square graph is used (remember four sub-blocks in an inch-square grid!). The total number of knots represented in every row of the column, called *columnar row total*, remains stable throughout the talim, e.g., 20 knots per columnar row. It could be more or less than 20 also, but remain so throughout if it is. The maintenance of this stability is the chief task of the talim writer. Thus, one column of a talim, which represents one-inch square of

[1] A short period was spent at Amritsar in 2015 to assess the talim usage there, but due to religious disturbances during the time of fieldwork, the work was suspended.

[2] Warp threads are vertically fixed on the loom and weft threads are those with which knots are woven, thus completing the horizontal structure.

[3] The developmental trajectory of graph and talim is discussed in a separate paper (under review) which also includes a contrast between manual and digital setting.

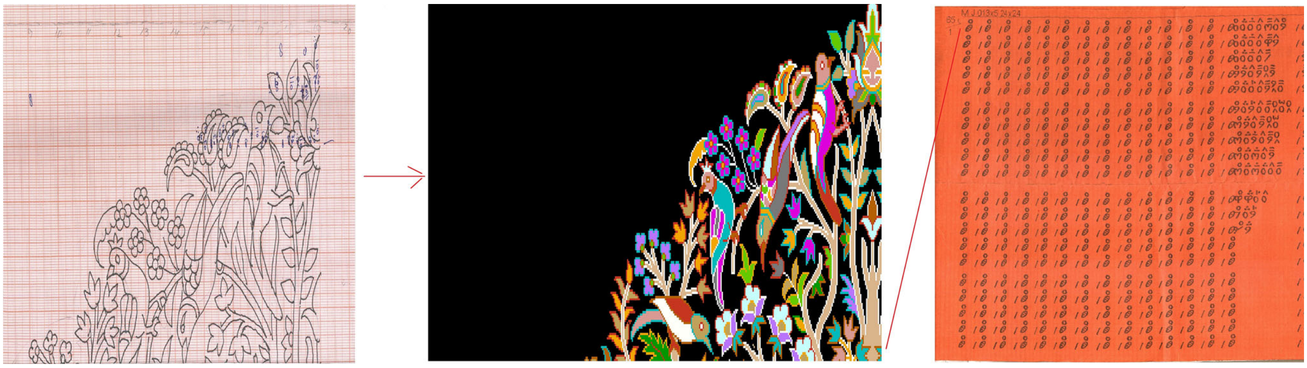


Fig. 1 A partially coded graph, its design and talim. Courtesy: Sajad Nazir, Srinagar. Since the designer digitally generated the talim after manually drawing the design by scanning it to the CAD system, this graph is partially coded for pedagogical purposes. Though it's a *portion* only of an actual carpet design, it will show the design progression

graph with 400 cells in it, represents 400 knots in the code, if its *columnar row* total is 20 knots.

A complete talim set may include *n* number of talim rolls depending on the size and the design type. One row in the talim represents weaving activity pertaining to one row of the carpet. Every roll is marked with a specific *page number*. Since it may not be possible to represent the code of one row of a longer-size carpet on one roll only, the additional paper strips are attached to the main roll, which are then called *parts* of that page. For instance, 21/3 means 3rd part of talim page no. 21 and is read accordingly by the weaver: first the main *page* and then the *parts*.

After the talim has been completely generated, a separate actor, called *talim copyist*, is employed to make its copies. It is required when the same carpet needs to be woven on more than one looms.

The design process culminates at this juncture which comprises two major cognitive activities: the design creation and the coding. Since the designer's work is incomplete without devising a mechanism for design communication, coding is considered here a part of design process, yet cognitively, these are two different activities which place differential cognitive demands on the actors: the designer needs to have a creative vision plus a fine calculative ability to work out the technical constraints, e.g., computing borders in relation to the central field, roping in innovative design features like shadow effects to bring some freshness in conventional designs, balancing tradition with novelty and design experimentation, etc., while a talim writer can proceed with calculative abilities solely as she needs to pick the codes already supplied to her. Accordingly, different cognitive dimensions belong to the artifacts used in both activities.

The talim is finally passed to the weaver (*kaalbaaf)* who may either weave directly from it by reading and interpreting the code herself, or by listening to it being read by the other weaver on the loom.

In a computer-assisted setting, the designer creates the design digitally. For instance, Naqash and Qaleen weavers, used in this practice, are CAD-based systems in which the designer is supplied with a coded color palette. Besides usual design features offered by a CAD software, these two offer a number of functionalities like scanning manually drawn graph designs in order to generate digital talim, printing only specific portions of talim, feeding antique talims back to the system and extracting the design out of it, creating designs of missing pages in a talim set, etc. Once the design is completed, the talim is generated by giving print command in as many copies, which, in principle, does away with manual generation of talim, and hence, the need of talim writers and talim copyists. The designer can do all these tasks herself in a digital setting. It is to be noted that computer-generated and handwritten talims are identical in nature, structure and function and are similarly processed by the weaver. See Fig. 2.

Despite digital revolution, however, manual setting, including code generation and copying, is still existent. The symbols used to compose the talim are (Tables 1, 2):

The color symbols are positioned either above or below the number symbol which is convention based, but may admit flexibility too. Once constructed, a typical talim roll looks like as follows (Fig. 3). Its different elements are indicated with arrows:

The talim is written or printed on usually orange, rust or brown-colored long paper strips. The roll is folded in the middle showing two blocks of columns at a time and is inserted in the warp threads of the loom. The instructions in the roll are then read from top to bottom, left to right and vice versa depending upon the design type embedded in the code (Fig. 4).

A Kashmiri carpet has distinct *design elements*. A talim set has code strips pertaining to following areas in design elements which are usually written on separate rolls:

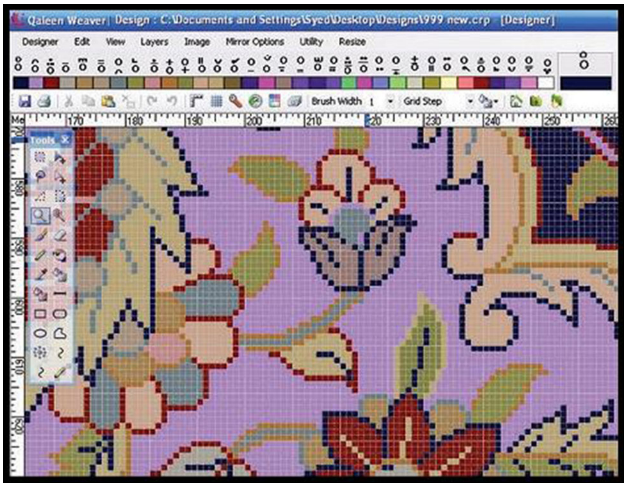

Qaleen Weaver | Design : C:\Documents and Settings\Syed\Desktop\Designs\999 new.crp [Designer]
Designer Edit View Layers Image Mirror Options Utility Resize
Brush Width 1
Grid Step
Tools

Fig. 2 Interface of Qaleen weaver. Courtesy: graphicsweave.com

Table 1 Numeral table

Numeral	Symbol
1	O
2	٩
3	M
4	φ
5	ϟ
6	/
7	7
8	ዋ
9	ɾ
10	Ø

Table 2 Color table (representative colors)

Color (English)	Color (in talim lexicon)	Symbol	Position [above/below the number symbol]
Black	Cheen	ɵ	above
White	Danti	∩	above
Yellow	Zard	M	above
Light yellow	Makai	U	above
Blue	Parozi	ʌ	below
Sky blue	Malie	—	Above
Green	Sabz	|·	above
Bottle green	Zangary	+	below
Light pink	Badami	∸	above
Dark pink	Gulabi	||	Above
Red	Anari	∴	above
Golden brown	Dalcheen	+	above
Dark brown	Doday	o~	above
Gray	Rackh	=	Above

1. borders, where lower border is taken as reference (*daul hashiya*),
2. angular corners, within borders beneath the *chauthai* (*kunjvaat*)
3. central field, including side borders, *chand* and *chauthai* (*mattan*).

Among these, the last forms the meat of the code. The code areas are shown in red font in the figure here. The different *design types,* as per the nature of repetition, are created then involving these design elements. The weave portion corresponding to first column of our example talim is shown as well. Note that, while the reading of talim goes top to bottom in the code, its weaving goes bottom to top on the loom.

Cognitive dimensions of talim

The cognitive dimensions (CDs) framework was formulated by Green (1989: 448) wherein he remarks that, ''cognitive dimension' of a notation is a characteristic of the way that information is structured and represented, one that is shared by many notations of different types, and by its interaction with the human cognitive architecture, has a strong influence on how people use the notation ….' The dimensions pertain to the structure of the representation and information represented in it. The CDs are a 'broad-brush assessment of almost any kind of cognitive artifact' (Green and Petre 1996: 131) which includes, 'mathematical expressions, diagrams, maps, timetables, human languages and of course music notation' (Blackwell et al. 2000: 1). It has been used for evaluating notations in design rational (Shum 1991), CAD systems (Petre and Green 1992), spreadsheets (Hendry and Green 1994), VPLs like LabVIEW and Prograph (Green and Petre 1996), music notations (Blackwell et al. 2000), a football simulation environment (Dagit et al. 2006), and even 'domestic devices' like telephones (Green 2000).

Blackwell et al. (2001) summarize 24 cognitive dimensions related to notations and information artifacts unearthed so far in various studies since Green (1989). The cognitive properties of talim are discussed with respect to this comprehensive framework. The CDs analysis of the second design artifact, i.e., the graph and its relation with talim, is reserved for a separate paper. A note on the terminology: the term *coder* refers to the actor generating the code manually. The talim is interchangeably referred to as *code* and the process of generating it as *coding*. While single strip of paper is called as *talim*, the complete set is called *talim set*. The total number of knots represented in one row of the column is termed as *columnar row total*. One complete row, from one end to the other end of the

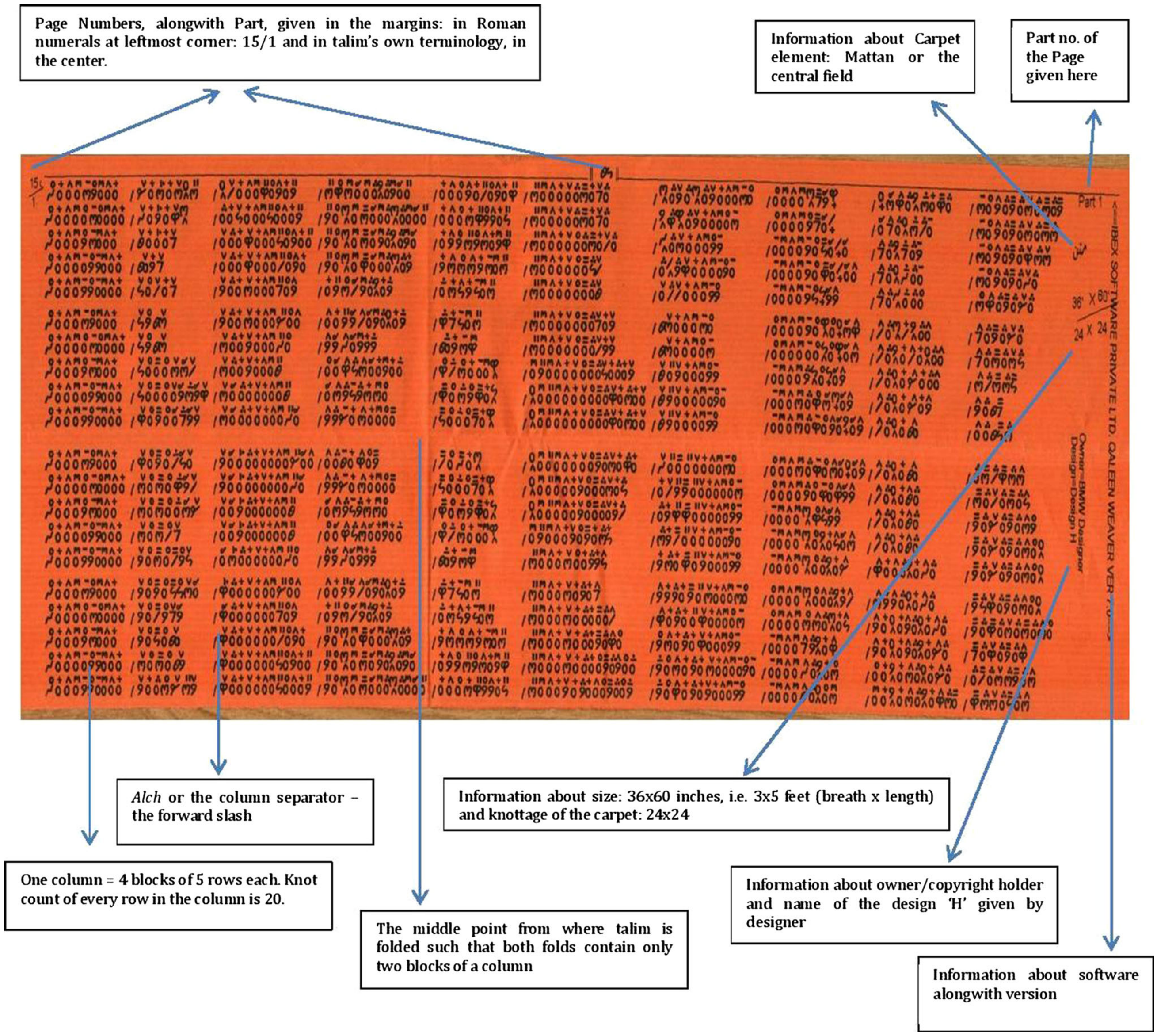

Page Numbers, alongwith Part, given in the margins: in Roman numerals at leftmost corner: 15/1 and in talim's own terminology, in the center.
Information about Carpet element: Mattan or the central field
Part no. of the Page given here
15/1
Part 1
36' X 60'
24 X 24
Owner=BMW Designer
Design=Design H
IBEX SOFTWARE PRIVATE LTD. QALEEN WEAVER VER
Alch or the column separator – the forward slash
Information about size: 36x60 inches, i.e. 3x5 feet (breath x length) and knottage of the carpet: 24x24
One column = 4 blocks of 5 rows each. Knot count of every row in the column is 20.
The middle point from where talim is folded such that both folds contain only two blocks of a column
Information about owner/copyright holder and name of the design 'H' given by designer
Information about software alongwith version

Fig. 3 Talim roll. Courtesy: M/s BMW Designers, Srinagar

talim, is one *instruction* that guides weaving activity of one row in the carpet. The designs drawn on graph sheets are simply referred to as *graphs*. Looking through CDs framework, the talim possesses following dimensions:

Viscosity

> How much effort is required to perform a single change? (Green and Petre 1996: 139)

The talim is less viscous as compared to programming notations like Labview (Green and Petre 1996) and Prolog (Green 1999) as local changes can be made easily in it. The errors generally pertain to *mis-calculation* of the knot cells from the graph or *mis-writing* the calculated knot cells in the columnar row. Both are detected on the completion of the current row being worked upon when the coder finds *columnar row* total to be more or less than the total being followed, e.g., if it is 20, it must be so in every columnar row, which implies that every row too will have a stable knot count. This structural division of code into columns and stable knot count aids the coder in identifying the errors: if not in the first column, then error might be in second and so on. The coder re-calculates the knot counts in every column and if any discrepancy is detected somewhere which, if it is due to:

1. mis-writing: then the error is rectified by shifting the respective knots in subsequent columns progressively, e.g., 5 g could have wrongly been written as 8 g. The talim, thus, displays knock-on viscosity.

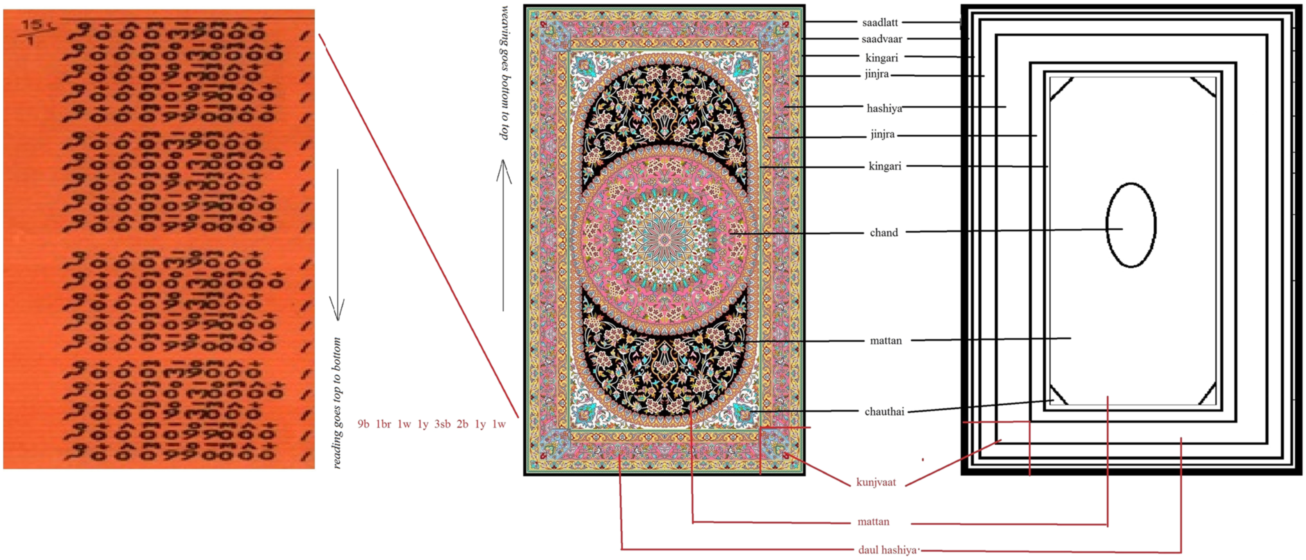


Fig. 4 Design elements and weave portion. Design and talim courtesy: M/s BMW Designers, Srinagar

2. mis-calculating: in which case, the coder examines the graph again to identify which cell knots had been counted wrongly, e.g., graph showed only three cells of green but 4 g was written in the talim.

Both cases may require rewriting of that particular roll to avoid overwriting. Thus, local changes are easier to make in talim without changing the overall notation, e.g., nothing needs to be changed in the previously written rows or rolls.

Visibility

> Is every part of the code simultaneously visible (assuming a large enough display) or is it at least possible to juxtapose any two parts side-by-side at will? If the code is dispersed, is at least possible to know in what order to read it? (Green and Petre 1996: 139)

The talim possesses high visibility as the code is written in a straightforward manner and comprises no hidden structures. However, at any time, only certain portion of the code is visible in both the coder's and the weaver's gaze: *the current talim roll* that the coder is writing or the one weaver is weaving, which she keeps inserted in the warp threads of the loom while weaving. If the code for the row under weaving spreads over a number of rolls, the weaver inserts these auxiliary rolls, indicated as *Parts*, beneath the main roll indicated as *Page*. Green and Petre (1996: 162) termed such an arrangement as '*juxtaposability*, the ability to see any two portions of the program on screen side-by-side at the same time.' The talim possesses this feature. The insertion of different *parts* of the same *page* in the warp threads of the loom gives weaver an opportunity to scrutinize the code dispersed over three different displays simultaneously.

Beyond what is currently inserted on the loom, the code of other portions remains concealed in other *Pages* and *Parts*. This dispersion of the code depends on the size and design of the carpet: the longer the carpet and non-repeating the pattern, the larger the number of pages required to write its code, which remains tied in loose bundles near the loom, from which the rolls are extracted one by one, are inserted in the loom, read, decoded and woven. The reading and decoding conventions are not stated anywhere in the code. These remain offscreen and differ from one design type to the other. For instance, in repeating design types, talim pertaining to only one unit of the repeatable motif/group of motifs is generated which is read in different combinations to weave the repetitions. This generation of code of a unit motif is akin to 'space saving heuristics' (Modungo et al. 1994: 102) and compresses the code. The talim's visibility is on par with spreadsheets (Tukianen 2001) and music scores (Nash 2015).

Premature commitment

> whether there are any constraints on the order of doing things. (Kutar et al. 2000: ix)

The work commitments differ from coder to weaver. In coder's case, the way codes are *picked* matches the way these are represented in the talim which is forward, sequential lining up. Because of this, the coder cannot move back or forth during writing and must adhere to the structure thrown by the code. For instance, she cannot start picking codes randomly from the graph. She must start from the bottommost row, proceed from left to right and deposit the codes in the talim in textual modality of top–bottom (T–B), left to right (L–R) only. No 'first marks' or separate declaration, as in VPLs (Green and Petre 1996), regarding eventual figure is done. However, she must adhere to the syntactic conventions of talim like writing codes in rows–columns, keeping columnar row total stable, ending columns with *alch*, etc. The coder must display this much 'commitment to the layout' (Green and Petre 1996: 156), but, unlike VPLs, need *not* show this commitment in the form of any 'first marks' on the roll display from where she starts writing the code.

Unlike coder, however, the weaver is not constrained to read the talim in L–R, T–B textual modality, even though the weaving proceeds in bottom–top (B–T) mode only on the loom. The weaver reads the instructions in different variations of L–R, R–L, T–B, B–T depending on the design type embedded in the code. Thus, the weaver though enjoys more freedom, but is still prematurely committed to the design type.

Hidden dependencies

> … relationship between two components such that one of them is dependent on the other, but that the dependency is not fully visible. (Green and Petre 1996: 153)

There are hidden dependencies in the talim in the form of decoding and hence weaving being dependent upon the kind of *design type* embedded in the code. Every design type requires different decoding convention on the part of weaver; for instance, a *kuldar* (half-vertical) pattern, where one half of the design is mirror repeated in the other half, requires decoding in the form of L–R for the left half and R–L for the second half of the same instruction. If the design type is not mentioned in the margins of the roll, the weaver will not come to know which decoding convention to be imposed on the code and consequently, she may misread the instruction leading to a distorted design on the loom. These decoding conventions are, however, not stated explicitly anywhere in the code and remain hidden.

Role expressiveness

> how clearly a 'chunk' of information structure conveys its functional role…. such that the visual appearance of the code can be used as a cue to identify its function (Shum 1991:338)

Perceptually, the entire code looks homogenous with color symbols likewise positioned above or below the number symbols throughout. Besides that, the two different sets of talims also look identical and can be easily mistaken for each other, like programming notations (Green 1999: 10). However, unlike these, the respective features of talim, namely the design name and type, the knottage, the copyright holder's name, etc., are indicated in the margins from which their differences are ascertained. However, there are lexical cues in the code whose role is readily inferred, e.g., the column separator, *alch* '/,' whose purpose is to make visual distinction between two columns and is written at the end of the columnar row, or another cue called *advaar* indicated by a '|' or '∼' which indicates onset of repetition in repeatable designs and is consequently, written at the end of the last column in the row. At this point, the weaver starts reading the instruction in reverse modality of R–L to weave the repetition. In our talim, this cue occurs at last column of 15/2. See this in Fig. 5.

Further, the division of overall code into *segments* pertaining to different design elements, i.e., borders (*hashiya*), central field (*mattan*) and border corners (*kunjvaat*), which are usually written on separate strips, also organizes the code. This segmentation works as a perceptual cue to infer the functionality of that particular roll. Thus, even though the talim possesses low role expressiveness, it has sufficient lexical and perceptual cues to counteract its homogeneity.

Error proneness

> Does the design of the notation induce 'careless mistakes'? (Green and Petre: 138)

Even though the talim uses a single specification scheme, it is highly vulnerable to errors and demands extreme patience and meticulousness on the part of the coder and the talim copyist, if employed. Since the coder needs counting every cell in the graph to generate the code and undertakes other mathematical operations to arrive at the final representation, this computing is a potential source of error. The high diffuseness of information in the graphs makes even simple counting cognitively challenging. Imagine counting every single cell in the graph representing 154,000 cells from a standard 22 × 17.5 inches graph sheet! The coder may miscalculate the cell knots, misdivide the total representation or simply mis-write the figure. The distribution of design over more than one graphs considerably increases the total cells to be 'picked,' thereby increasing the cognitive load of the coder by placing high demands on her working memory. Naturally, it takes coders weeks to generate the complete code where misrepresentation of a few knots can ruin the overall design. The talim copyists may further aggravate this stress by simply mis-copying. Harris (2003: 3) notes that 'the commitment to making a copy is exacting—a handful of mistakes in 100 lines of texts can make a design unusable.' The talim coding is certainly more error prone than comparable musical score creation (Nash 2015)

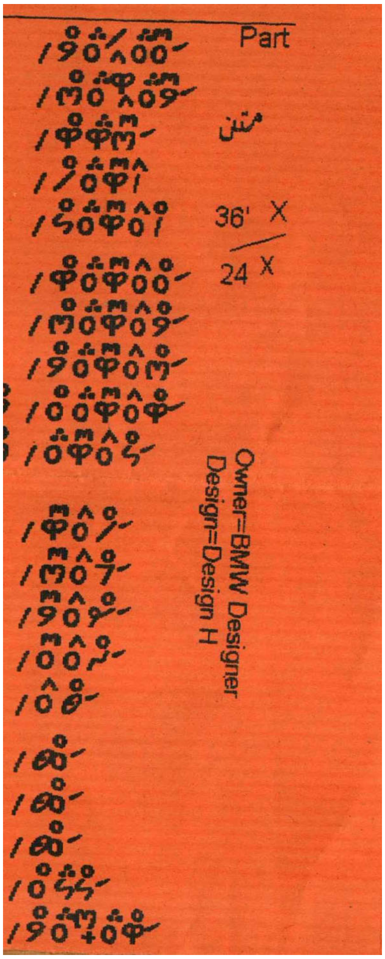


Fig. 5 Last column showing the advaar indicators

Abstraction

> An abstraction is a grouping of elements to be treated as one entity, whether just for convenience or to change the conceptual structure' (Green and Petre 1996: 144)

The *design type* and *design elements* are the only abstractions in the talim. One design type imbibes a particular mode of creating and interpreting the code which differs from the other design type. The design of a carpet is divisible into quarters or halves to create repeatable patterns in them, which are called here *design types*. For instance, in a *kashyan* design type, the design depicted in one quarter is mirror repeated in the other three quarters. Consequently, to save the cognitive effort involved in both drawing the design and writing the code, the design pertaining to only one quarter is created and the talim accordingly for only one quarter is generated. The rolls are labeled as *kashyan*, and the weavers interpret them from left to right (1st quarter), right to left (2nd quarter), last page first left to right (upper 3rd quarter) and last page first right to left (upper 4th quarter) to weave design on the carpet. The labeling of the rolls as 'Kashyan' indicates this abstraction. If the rolls are mistakenly labeled as *kuldar* where one vertical half of the design is mirror repeated in the other half, the rolls will be interpreted as per that design type, leading to distortion in the design during weaving. However, this needs noting that, at times, the information about design type is conveyed only verbally to the weaver.

Next, the *grouping* offered by *design elements* also provides an abstraction mechanism. Recall that, the design pertaining to the different design elements, i.e., borders, corners and central field, is usually created separately representing only a unit of repeatable motifs in them and consequently, separate code generation for these groups. The unit code for these units of repeatable motifs is an abstraction that hovers over the missing portions and enable their creation via interpretation by the weavers. All these are *persistent* abstractions (Green and Blackwell 1998: 26).

Secondary notation

> Use of 'layout, colour, other cues to convey extra meaning, above and beyond the "official" semantics of the language' (Green and Petre 1996: 139)

The talim contains a lot of useful secondary notation, which is though not part of the executable portion of the code, but rather work as a sub-device in the system.[4] This information, given on the margins of the talim rolls, pertains to page numbers and their respective parts, design type, design name, copyright holder's name, design element, software's name and version in digital setting, etc. Some of this is indispensable and regulates the weaving process, i.e., the page number and the part, which are given in Roman in corners and in talim lexicon in the middle of the roll. If page-specific information is absent, the weaver will not know where to weave that particular roll, whereas the information on design element is both regulative and indicative, e.g., 'mattan' written in the margins indicates that weaving pertaining to central field is going on. In contrast, some information is only indicative, e.g., designer's name. This informational content is akin to indicative 'comments' in programming notations like HTML (Green et al. 2006) and regulative comments in music sequencers (Blackwell et al. 2000).

[4] I am grateful to the anonymous reviewer for pointing this out.

Closeness of mapping

> How closely do elements on the display relate to the searcher's problem domain? (Hendry and Harper 1996:1038)

The talim relates differently to different actors. The coder uses it *in making* during code generation. While picking codes, the data flow from L to R in the graph, which she, as it is, sequentially deposits in the talim: total representations, one by one. Thus, the structure of the code maps the coder's goals: sequential deposition of the coded data. On the other hand, the single problem that a weaver faces is: which *number of knots* and *in which color* should be woven so that the design emerges in the end? The talim directly addresses this problem: it guides this weave action in the same way the weaving should proceed. For instance, if four knots of green need followed by five knots of blue to create the design, it will be exactly represented in talim as: 4g 5b. The talim mirrors the cognitive understanding of the weaving process by the weaver: how *she* sees the problem and its solution. Rephrasing Green and Petre (1996: 147), 'the objects in the programming domain … match the behaviour of the objects in the problem domain…' or, there occurs 'behavioural similarities between the symbols in a notation and their referents' (Dearden et al. 2003: 383).The closest that comes in this context are the musical scores (Nash 2015).

Consistency

> whether similar semantics are expressed in similar syntactic forms. (Kutar et al. 2000: ix)

The homogenous idea conveyed by talim, i.e., number of knots to be woven in a particular color, is categorically represented as: *color code, written above or below, the number code*. There is no other way of expressing this idea. The talim is highly consistent in this matter which imparts lower diffuseness, non-ambiguity and high visibility to the code.

Diffuseness

> Some notations use a lot of symbols or a lot of space to achieve the results that other notations achieve more compactly' (Green and Petre 1996: 148)

The talim uses only two symbols to convey its principle idea of number of knots falling under a particular color: the *color symbol* positioned above or below the *number symbol*. Even if, these two symbol types branch out into a number of tokens (say 35 color symbols and 30 number symbols, if columnar row total is taken as high as that), one unit of talim's instruction comprises of two symbols only. Thus, the talim is a quite compact notation, less diffused as compared to programming notations like Prograph which require a number of distinct entities for constants, connectors and so on (Green and Petre 1996: 149). One reason for less diffuseness is the greater *closeness of mapping* in talim, as Green and Petre (1996: 148) observe that the, 'notations that have a very close mapping to the problem domain will require fewer lexemes to achieve their results and will therefore appear terse.' However, insofar as spatial distribution of informational content in *entire talim set* is concerned, it depends upon the carpet size and the design type embedded in the code. The lower carpet size warrants lesser space (say 20 pages only), but larger carpet sizes may require larger space (more than 100 pages). The repeatable patterns further reduce the overall diffuseness of the code, with only representative unit being generated in the design as well as in the talim.

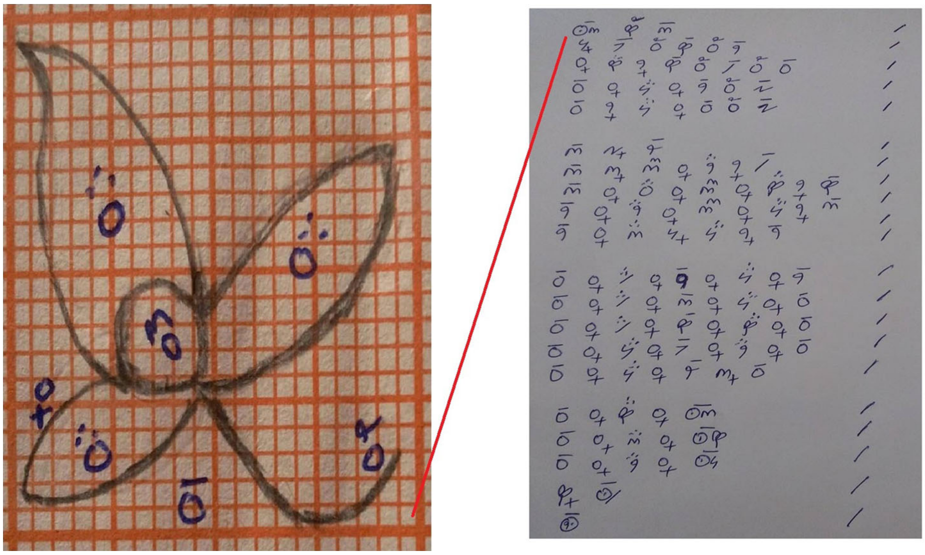
Hard mental operations

The talim is cognitively taxing to write. The coding requires counting the cells falling under a particular color code in the graph, converting the count to its symbolic representation and writing the same in the columns. If the columnar row total in that column exceeds this count, then it is divided into the current and the subsequent column. For instance, in a columnar row total of 20 knots, if the representation so far has proceeded as:

> 5w, 3b, 3g, 6r [5 white, 3 blue, 3 green, 6 red] = 17 knots
> and the next that the coder encounters is eight cells of yellow, how should she accommodate the same within this row? Writing 8y would exceed the columnar row total to 25, thereby bringing instability into the structure (Fig. 6). To avoid this, 8y is distributed among two columns and the resultant instruction becomes:
> 5w, 3b, 3g, 6r, 3y/5y,……

The process can be formalized as follows. Proceeding from bottom-left corner of the graph, the coder proceeds left to right, and

1. count the knot cells falling under a particular color code
2. write the representative figure (F) in current column (C1)
3. keep tab of columnar row total (T),
4. if F > T, then
5. subtract F − T = F1
6. write F1 in C1
7. and rest of F in the next column (C2)



13sb 4br 3sb /
5g 7sb 1br 4sb 1br 2sb /
1g 4r 2g 4sb 1br 6sb 1br 1sb /
1sb 1g 5r 1g 2sb 1br 9sb /
1sb 2g 5r 1g 1sb 1br 9sb /

3sb 9g 8sb /
3sb 3g 3y 1g 2r 2g 6sb /
3sb 1g 1r 1g 3y 1g 4r 2g 4sb /
2sb 1g 2r 1g 3y 1g 5r 2g 3sb /
2sb 1g 3r 5g 5r 2g 2sb /

1sb 1g 6r 1g 2sb 1g 5r 1g 2sb /
1sb 1g 6r 1g 3sb 1g 5r 1g 1sb /
1sb 1g 6r 1g 4sb 1g 4r 1g 1sb /
1sb 1g 5r 1g 7sb 1g 2r 1g 1sb /
1sb 1g 5r 1g 8sb 3g 1sb /

1sb 1g 4r 1g 13sb /
1sb 1g 3r 1g 14sb /
1sb 1g 2r 1g 15sb /
4g 16sb /
20sb /

Fig. 6 Sample design unit designed, coded and transcribed by researcher

Thus, the coding comprises numerous cognitive operations like calculation, comparison, subtraction and inference which the coder goes about in this very sequence without representing it anywhere and comes up with the eventual representation, i.e., F, all of which place extreme burden on her working memory. Computing cells from more than one graphs put further cognitive load. Green and Petre (1996: 138) ask, 'Are there places where the user needs to resort to fingers or penciled annotation to keep track of what's happening?' During coding, the coder primarily makes use of finger or pen-based calculation as she proceeds by pointing the cells with her pen on the graph, counts the cells sequentially by moving the pen along the uncounted cells, calculates and writes the final figure on the roll. Without consistent pen-based support, it is virtually impossible to spot and count tiny cells inside the grid as pointing with a pen cognitively fixes the cells during computation. However, except eventual figure, the intermediary totals are not represented anywhere. Due to technicalities like above involved in code generation, talim involves, what Nash (2015:194) calls a strong 'literacy threshold' as it requires special training on the part of the novices and the lay readers to comprehend the code. For an untrained eye, the talim is just a scribble in a strange script, just like a musical score.

Progressive evaluation

> Can a partially-complete program be executed to obtain feedback on 'How am I doing? Cox (2000: 101)

The coder can evaluate her progress during coding courtesy the columnar structure and the stable knot count. On completion of the row, she can check whether knot count in every columnar row is stable and more or less counts are corrected accordingly. The constant evaluation and easier modification reduce viscosity in the notation. Further, the weaver need not wait to complete code generation to start weaving. Since the code for different design elements is generally represented on separate strips, the ones already finished can be passed onto the weaver for weaving, while the coder continues generating code for other portions. This parallel processing saves considerable time. However, whether the design encoded *is actually workable* or not, that can be assessed only when the weaver actually weaves it, not before that. This is because the design cannot be inferred from the symbols alone—a fact noted for shawl weaving by Harris (2003) as well.

Creative ambiguity

> extent to which a notation encourages or enables the user to see something different when looking at it a second time. (Blackwell et al. 2001: 335)

The talim is non-ambiguous as symbols represent distinct referents. Its low diffuseness and precise representation restrict alternate interpretations by the reader. Over longer periods, however, the conventionality may be compromised as new symbols keep being invented or the meanings of existing symbols get altered. The older talims, consequently, may invite different interpretations.

Specificity

> The notation uses elements that have a limited number of potential meanings… rather than a wide range of conventional uses… (Blackwell et al. 2001: 335)

The talim symbols have limited meanings, even though these may change over longer time periods. The actors may not recognize a referent of a totally obsolete symbol, a newly invented one, or even correct referent of an existing symbol whose referent has been deliberately changed by the designer, but *it is unlikely* that inter-code confounding is possible: that is, since color codes are always positioned *above* or *below* the number code, it is unlikely that *these* could be mistaken for number codes and vice versa. This *specificity* involved in color number representation, restricts their potential meanings to a great extent and eases their processing by the actors (Stenning and Oberlander 1995).

Details in context

> how elements relate to others within the same notational layer (Blackwell et al. 2001: 335)

The symbols do not bear any causal or interlinking relation with other symbols in the instruction as their arrangement is sequential depositing.

Indexing

> notation includes elements to help the user find specific parts (Blackwell et al. 2001: 335)

There is no procedure to search either a particular unit or a weave pattern in the code. Though the codes of particular *design elements* can be located courtesy their labeling in the margins of the rolls, e.g., the label *daul hashiya* refers to code pertaining to lower borders of the carpet within the set, etc.

Synopsis

> provide an understanding of the whole when you 'stand back and look' (Blackwell et al. 2000: 335)

It is not possible to have a 'gestalt view' (Whitley and Blackwell 1997) of the talim. Even if rolls are spread physically, one after the other, on the floor, no information about design embedded in it can be obtained.

However, in certain cases, a conjecture about the *shape* of the carpet can be made. The carpets have canonically rectangular shapes, but in experimental ones like round carpets, the code itself may give an idea about circularity. For such shapes, the coders usually start writing the code from *middle portion* of the roll leaving column spaces on both sides blank and progressively increase the code area in middle from both sides on subsequent rolls. For instance, if Page-1 represents two columns of the code right in the middle of the strip, Page-2 may represent four columns of code in the middle, followed by six columns on Page-3 and so on. The coder may also choose to write *false codes* in the blank columns. These false codes are *not to be* woven so that the warp threads pertaining to these code numbers can be left unwoven, and make the carpet woven from the middle portion on the loom, whose codes are represented in the middle portion on the roll. For instance, in a *non-repetitive design*, instructions could be written as:

> 24** / 24** / 24** / 4g 8b 8y 4w / 24** / 24** / 24** [columnar row total = 24]
> 24** / 24** / 3y 8b 7g 2w 4b / 2y 9b 5g 4b 4w / 3g 6w 6g 5y 1b 4w / 24** / 24** /

Here, the weaver is enjoined to interpret the false codes of '**' as *non-weaving knots,* upon which she leaves 72 warp threads on the left side on the loom and starts weaving with 4g in the middle. Similarly, after weaving 4w, she again leaves 72 warp threads on the right. As the weaving area increases from middle in the later rows, the round shape of the carpet starts emerging on the loom. While talim rolls for such shapes can separately also give the idea about shape of the carpet embedded in it, a gestalt view offered by juxtaposing the rolls one after the other let the reader observe this circularity in the progressive unfolding of the code (indicated in blue font) from the middle, as can be observed in above two rows. It is to be noted that above is the *simplest example* of such coding scheme which differs in a *repetitive* pattern in which the repetitive codes, at the right side, are necessarily omitted and are replaced by *advaar* at the end of the roll.

Permissiveness

> notation allows several different ways of doing things (Blackwell et al. 2000: 336)

The homogenous idea represented by talim, i.e., number of knots falling under a particular color, can be represented only in a linear manner, i.e., number of knots in a color, where this unit is indicated by symbol and color codes. There is no other way of representing this idea.

Lability

> notation changes shape easily (Blackwell et al. 2000: 336)

The talim remains consistent throughout and does not change shape.

The other dimensions, viz. *provisionality, useful awkwardness, unevenness* and *free rides*, are not relevant to the talim. The trade-offs between different dimensions can be observed in the case of talim as well. Since talim does not have any *hidden dependencies*, it is easier to make local changes in it without changing the overall notation. That is, it displays lower *viscosity*, unlike in programming languages which are highly viscous due to larger number of dependencies in them. This fact and lower *diffuseness* render higher *visibility* to the code. The *secondary notation* too reduces complexity and organizes the code better.

Conclusion

This paper described cognitive dimensions possessed by a coded script, called talim, used in Kashmiri carpet weaving practice through CDs framework of Green (1989) and Blackwell et al. (2000). Green (1989: 446) remarks that 'Each notation highlights some types of information at the expense of obscuring other types; each notation facilitates some operations at the expense of making others harder. *A notation is never absolutely good, therefore, but good only in relation to certain tasks*' [italics original].

This is certainly true of talim. If contrasted with graph from which it is generated, some information is displayed by graph but not talim and vice versa. The graph gives a visual conjecture about design, but not talim, while talim guides the weaving activity, but not the graph. Imagine if weaver were to extract all information from a densely designed and coded graph! The talim eases this cognitive load by giving precise computations which may involve hard mental operations by the coder, but it organizes the weaving activity of the weaver. As an external memory resource, it allows cognitive offloading of weaving patterns, which, otherwise, the weaver had to memorize. Evidently, it performs differently at the hands of different actors. This shows the emergence of cognitive dimensions to be function of the task and the user. The different tasks may reveal different dimensions to different users, which is why perhaps Green (1989) remarks that no dimensions are good or bad in themselves.

Acknowledgements I am thankful to National Institute of Advanced Studies (NIAS), Bengaluru, and its Consciousness Studies Programme for supporting this work and funding the fieldwork conducted during 2015. I am obliged to Ms. Aamina Assad, Chief Designer, School of Designs (SoD), Mr. Gazanfar Ali, the then Director, Directorate of Handicrafts—Massive Carpets Scheme (MCS), Mr. Zubair Ahmad, Director, Indian Institute of Carpet Technology (IICT), all in Srinagar, for facilitating my work at their respective institutions. The *consent form* used during 2015 was translated by Prof. Mushtak Haider, University of Kashmir. I am thankful to M/s BMW Designers and Sajad Nazir, Srinagar, for permitting me to reproduce the talim roll, a talim excerpt, graph and their respective designs in the paper whose copyright they hold. I am indebted to Prof. Siby George, IIT Bombay, for giving me feedback on the paper. I am grateful to all my respondents for giving me their invaluable time.

References

Blackwell AF, Green TRG, Nunn DJE (2000) Cognitive dimensions and musical notation systems. Workshop on notation and music information retrieval in the computer age (ICMC 2000), Berlin

Blackwell AF, Britton C, Cox A, Green TRG, Gurr C, Kadoda G, Kutar MS, Loomes M, Nehaniv CL, Petre M, Roast C, Roe C, Wong A, Young RM (2001) Cognitive dimensions of notations: design tools for cognitive technology. In: Beynon M, Nehaniv CL, Dautenhahn K (eds) Cognitive technology: instruments of mind - CT 2001, LNAI 2117. Springer, Berlin, pp 325–341

Cox K (2000) Cognitive dimensions of use cases - feedback from a student questionnaire. In: Blackwell AF and Bilotta E (eds) Proceedings of Psychology of Programming Interest Group, (PPIG-12), Cozenza

Dagit J, Lawrence J, Neumann C, Burnett M, Metoyer R, Adams S (2006) Using cognitive dimensions: advice from the trenches. J Vis Lang Comput 17(4):302–327

Dearden A, Siddiqi J, Naghsh A (2003) Using cognitive dimensions to compare prototyping techniques. In: Proceedings of 15th annual workshop of the Psychology of Programming Interest Group (PPIG 2003)

Gans-Reudin E (1984) Indian carpets: with 280 illustrations, 120 in colour. Thames and Hudson, London

Goswami KK (2009) Developments in handmade carpets: an introduction. In: Goswami KK (ed) Advances in carpet manufacture. Woodhead Publishing Ltd., Oxford, pp 138–181

Green TRG (1989) Cognitive dimensions of notations. In: Sutcliffe VA, Macaulay L (eds) People and computers. Cambridge University Press, Cambridge, pp 443–460

Green TRG (1999) Building and manipulating complex information structures: issues in prolog programming. In: Brna P, du Boulay B, Pain H (eds) Learning to build and comprehend complex information structures: prolog as a case study. Ablex Publishing, New york, pp 7–27

Green TRG (2000) Instructions and descriptions: some cognitive aspects of programming and similar activities.In: Proceedings of working conference on Advanced Visual Interfaces (AVI 2000), ACM Press, New York, pp 21–28

Green TRG, Blackwell AF (1998) Cognitive dimensions of information artefacts: a tutorial. Available online at http://iihm.imag.fr/blanch/ens/2010-2011/M1/EIHM/cours/1998-Green-CognitiveDimensions.pdf

Green TRG, Petre M (1996) Usability analysis of visual programming environments: a 'cognitive dimensions' framework. J Vis Lang Comput 7:131–174

Green TRG, Blandford AE, Church L, Roast CR, Clarke S (2006) Cognitive dimensions: achievements, new directions, and open questions. J Vis Lang Comput 17:328–365

Harris P (1991) The Kashmir shawl: lessons in history and studies in technology. Ars Textrina 16:105–127

Harris P (2000) Decoding the talim. Hali 110:82–83

Harris P (2001) Kashmiri Shawl Survival. Textile Forum, No. 3 August

Harris P (2003) Digital images in Kashmir shawl weaving. Complex Weav J 71

Harris P (2007) An eighteenth century digital technology. Cahiers métiers d'art*Craft J 1(1). Available online at http://tapadesi.com/published-articles/

Hendry DG, Green TRG (1994) Creating, comprehending and explaining spreadsheets: a cognitive interpretation of what discretionary users think of the spreadsheet model. Int J Hum Comput Stud 40:1033–1065

Hendry DG, Harper DJ (1996) An informal information-seeking environment. J Am Soc Inf Sci 48(11):1036–1048

Kutar M, Britton C, Wilson J (2000) Cognitive dimensions – an experience report. In: Blackwell AF, Bilotta E (eds) Proceeding of the 12th annual meeting of the Psychology of Programming Interest Group (PPIG 2000), Italy, pp 81–98

Lawrence W (1895) The valley of Kashmir. Oxford University Press, London

Leitner GW (1882) Linguistic fragments discovered in 1870, 1872 and 1879 relating to the dialect of the magadds, and other wandering tribes, the argots of thieves, the secret of trade-dialects and systems of native cryptography in Kabul, Kashmir and the Punjab followed by an account of shawl-weaving. Punjab Government Civil Secretariat Press, Lahore

Mathur AR (2004) Indian carpets: a hand-knotted heritage. Rupa & Co., New Delhi

Modungo F, Green TRG, Myers BA (1994) Visual programming in a visual domain: a case study of cognitive dimensions. In: Cockton G, Draper SW, Weir GRS (eds) Peoples and computers IX: proceedings of HCI'94. Cambridge University Press, Cambridge, pp 91–108

Moorcroft W, Trebeck G (1841) Travels in the Himalayan Provinces of Hindustan and the Punjab, Ladakh and Kashmir; in Peshawar, Kabul, Kunduz and Bokhara: From 1819 to 1825, vol 2. John Murray, London

Nash C (2015) The cognitive dimensions of music notations.In: International conference on technologies for music notation and representation (TENOR-2015), France, pp 191–203. Available online at http://tenor2015.tenor-conference.org/TENOR2015-Proceedings.pdf

Petre M, Green TRG (1992) Requirements of graphical notations for professional users: electronics CAD systems as a case study. Le Travail Humain 55(1):47–70

Roy T (2004) Traditional industry in the economy of colonial india. Cambridge University Press, Cambridge

Sajnani M (2001) Encyclopaedia of Tourism Resources in India, Vol. 1.Gyan Publishing House, India

Saraf DN (1987) Arts and crafts, Jammu and Kashmir: land, people, culture. Abhinav Publications, New Delhi

Shum S (1991) Cognitive dimensions of design rationale. In: Diaper D, Hammond NV (eds) People and computers VI: proceedings of HCI'91. Cambridge University Press, Cambridge, pp 331–344

Stenning K, Oberlander J (1995) A cognitive theory of graphical and linguistic reasoning: logic and implementation. Cogn Sci 19(1):97–140

Tukianen M (2001) Evaluation of the cognitive dimensions questionnaire and some thoughts about the cognitive dimensions of spreadsheet calculation. In: Kadoda G (ed) Proceedings of 13th annual workshop on Psychology of Programming (PPIG-13), pp 291–301

Whitley KN, Blackwell AF (1997) Visual programming: the outlook from academia and industry. In: Proceeding of ESP '97, 7th workshop on empirical studies of programmers, pp 180–208