ISSN 1744-1986

Technical Report N° 2010/ 16

Give us a clue!

Jon G. Hall and Lucia Rapanotti

06 October, 2010

Department of Computing
Faculty of Mathematics, Computing and Technology
The Open University

Walton Hall, Milton Keynes, MK7 6AA
United Kingdom

http://computing.open.ac.uk

Give us a clue!

Jon G. Hall Lucia Rapanotti
The Open University
{jon,lucia}@solvemehappy.com

1. Welcome odd dignitary on top form to middle-of-road club, it's said (12)

For some time now, we have been investigating new thought tools for problem solving, through our problem solving calculus. The problem class that we have most experience with are engineering problems, a class that includes mission-critical systems, business process reengineering, the design of educational experiences, of seating plans in open plan offices, etc.

As any student of Torquemada, of Ximenes, of Araucaria will tell you there are other problem classes, not necessarily mission-critical, that might interest the problem solver not least in the study of their methods of solution.

Indeed, cryptic crosswords provide some of the most intensely satisfying problem solving experiences: filling in that last solution is a crowning achievement to any day (no matter how seldom it occurs). Cryptic crosswords are an admix of wordplay and logic, they require few resources, and have little mission-critical importance.[1] In this sense, they are as far from engineering problem solving as can be.

So, in this article, we apply our problem solving calculus in the new setting, that of cryptic crosswords.

This is not a frivolous act: Our intention is to gather evidence that our calculus provides a generic approach to problem solving that spans diverse application areas. Moreover, that crosswords provide the quintessential form of tangled problem, we hope further to be able to explore how problems that tangle can be solved.

It is worth adding, however, that nothing we do in this article will make cryptic crosswords easier to solve.

For diversion, section titles are in the form of cryptic clues.

2. Obscure burial place found with chip (7)

The fun in solving a cryptic crossword exists in solving the wordplay that defines the puzzle part. In particular, much of the problem solving activity happens in identifying the structure of

1. Other than to the setter.

the clue as was intended by the setter. The wordplay involved is very often in the decoding of structure indicators. Of course, there is some variation from setter to setter, but many resources exist that give the general idea of how structure indicators work. In this article, we use those from a comprehensive web resource[2] that lists 43 such indicators grouped into 14 indicator classes.

The classes include:

- the *anagram indicators*, such as that used in the phrase 'broken carthorse', in which 'broken' indicates that 'carthorse' should be considered an anagram (one such being 'orchestra'). As it is written on the left, this is a *left anagram indicator*; there are *right* and *middle anagram indicators* too;
- the *hidden word indicators*, such as that used in the phrase 'within more lice', in which 'within' indicates a hidden word (one such being, 'relic'). Again, there are *left* and *right hidden word indicators*;
- the *alternate letters indicators*, such as that used in the phrase 'oddly happy', in which 'oddly' indicates we should consider only the odd letters of 'happy', so to give 'hpy'.

The professional setter will present the solver with complex – most often ambiguous – combinations of indicators, making full use of the fact that indicators may apply in various ways within the same clue: that 'broken' is a left and not a right indicator in a clue is determined only contextually, that is, the structure of a clue is not necessarily uniquely determined. Words or phrases in a clue might be indicators themselves, or the subject of an indicator.

To help the solver, part of a cryptic clue will be a defining clause with which the solution to the clue will agree. This most often is written either at the beginning or the end of the clue, perhaps separated by a comma, 'to', 'giving', or some other connective.

Clue solving, as a process, involves many different skills: the identification of the defining clause, the recognition of indicators, the resolution of ambiguity, clever guesswork, and making best use of other partially solved clues. By its nature, it can be interrupted – another clue catches your fancy, a train journey completes – and so it is worthwhile being able to annotate a clue with the state of the solving process. To this end, we introduce a simple clue markup language, SCMLq , that allows us to indicate our current knowledge of a clue's structure.

2.1. A Simple Clue Markup Language

There are but a small number of annotations in the markup. The first allows the definition part of a clue to be distinguished through a double underline.

Example: To indicate the definitional clause of the clue, 'Obscure burial place found with chip', we would write '$\underline{\underline{\text{Obscure}}}$ burial place found with chip' to indicate that we think the solution will mean 'Obscure'.

2. crossword tools: http://www.crosswordtools.com/cm/

The second annotation uses a single underline to identify an indicator, adding a subscript to indicate its class, if not otherwise obvious. We do not constrain the choice of subscript for an indicator class: that most redolent to the solver is the most appropriate. Parentheses are used to delimit the word or phrase to which it applies, if any, and between two scoped indicators goes a semi-colon.

Example: In the case of 'broken carthorse', for instance, we would write '$\underline{\text{broken}_\text{a}}$(carthorse)': 'broken' is an anagram indicator (subscript 'a'), and its scope is 'carthorse'. We might also omit the subscript in this case. A right indicator is written postfix, so that 'the messy' becomes '(the)$\underline{\text{messy}_\text{a}}$'.

Included in this case is when an indicator is itself the subject.

Example: 'chip' often indicates the abbreviation 'ic'; we might write '$\underline{\text{chip}}$'. In the clause 'chip away', we may annotate it '$\underline{\text{chip}}$; away' or '$\underline{\text{chip}_\text{p}}$(away)' for the two different parses (the latter indicating 'part of').

In designing the *SCML*, we have considered it most important that the clue can be annotated in place in the crossword; our recommendation is to use a pencil and to keep a good plastic eraser handy.

Example: The markup of the cryptic clue

> Note the shuddering domestic appliance Bill regularly installed, noisy thing

might arrive from the following analysis:

- 'Note' often indicates a musical note, resolving to one of 'a' to 'g', 'do', 're', 'mi', etc;
- 'the shuddering' may be an anagram indicator applied to 'the';
- the 'regularly' of 'Bill regularly' may indicate alternate letters ('t'); i.e., 'bl' or 'il'; and
- 'installed' suggests the embedding ('e') of those letters within something meaning 'domestic appliance'.

All of this would leave 'noisy thing' as the defining clause. The marked-up clue would then be:

$$\underline{\text{Note}};\text{(the)}\underline{\text{shuddering}_\text{a}};\text{(domestic appliance,(Bill)}\underline{\text{regularly}_\text{t}}\text{)}\underline{\text{installed}_\text{e}};\underline{\underline{\text{noisy thing}}}\ (6,7)$$

What has gone before is crossword specific. Now, armed with the *SCML*, we can turn to our research topic.

3. Topsy-turvy Eastern conundrum excludes non-ferrous metal, it's said, for our research topic (7,11)

Our problem solving calculus, based on problem orientation (PO), considers problems as first class objects, and defines the ways in which one problem can be transformed into one or more problems. Sometimes these transformations make a problem easier to solve. It has been defined through its application to – initially – software problems, including those of requirements engineering and, latterly, general engineering problems. Published studies include: hardware software co-design in a safety critical setting ([1]), business process reengineering ([2]), pattern oriented software development ([3]), the development of educational materials and environments ([4]), and the planning of seating arrangement in an open plan office[3]. In preparation are works on risk visualisation, and a meta-framework for the description of IT Governance frameworks.

Our calculus is based on a problem oriented interpretation of Rogers' definition of engineering ([5]) that is suited for reasoning: hence Problem Oriented Engineering (POE). At the centre of the interpretation is the problem form we refer to as an *Environment/Need/Solution (ENS) triple*. Rogers' defined engineering as

> the practice of organising the design and construction of any artifice which transforms the physical world around us to meet some recognised need

For *artifice* read Solution; for *physical world around us* read Environment; for *recognised need* read Need. Environment and Need pertain to the problem space whereas Solution lives in the solution space. An engineering problem is captured in the problem solving calculus as an ENS-triple.

Cryptic clues are not engineering, and – as problems – have a different structure. For them the problem space is the world of wordplay and logic; the solution space is the world of words and phrases. More formally, and as with all problems, cryptic clues consist of a problem part and a solution part; but, unlike engineering problems, the problem part of a cryptic clue is a triple:

- the identifier[4] (such as '12 across'),
- the puzzle part (such as 'Obscure burial place found with chip'), and
- the letter count (written parenthesised; here '(7)').

So, for us, the problem-part P of a problem is the triple of a Clue $C = (Id, Clue, Count)$. The solution part, S, is, more simply, a word or phrase.

3.1. Solving a clue

Early engineering problem solving under POE is a form of understanding which leads to a workable problem description[5]. In contrast, a precise initial description of the problem – the clue – is given; the solver thus begins by interpreting the clue to determine its wordplay structure.

Of course, we would like the solution to be the *correct* solution[6]. We will say that word or phrase W solves clue C, written W *solves* C, whenever W is the word or phrase that was intended by

3. As yet unpublished, but see http://problemoriented.wikispaces.com for a slide show on the approach.
4. We will sometimes elide the clue identifier for brevity in this paper.
5. The details can be complex: see [6].
6. The engineering form of a problem has a similar notion of correctness which reflects Rogers' use of the word 'meet' in his definition; the notion of correctness for an engineering problem is adequate wrt all stakeholders.

Clue	Justification
'Note the shuddering domestic appliance Bill regularly installed, noisy thing'	
'Note the shuddering domestic appliance Bill regularly installed, noisy thing'	A comma often separates definition and subsidiary parts.
'Note;the shuddering domestic appliance;(Bill)regularly$_{t}$;installed;noisy thing'	'Note' often indicates a musical note. i.e., one of 'a-g', or 'do-ti', or variants; 'regularly' is an alternate letter indicator
'Note;(the)shuddering$_{a}$;(domestic appliance,(Bill)regularly$_{t}$)installed$_{e}$;noisy thing'	'Installed' is an embedding indicator; 'shuddering' is an anagram indicator We have a complete explanation of the clue's structure!

Table 1: The problem exploration for: 'Note the shuddering domestic appliance Bill regularly installed, noisy thing'

the crossword setter to solve clue *C*: the setter is the arbitrator of correctness; how could it be otherwise?

(There is a weaker version of solves in which we allow any solution which is indistinguishable from the crossword setter's intentions, but which isn't their solution. The clue

Start golf drink? (3)

is satisfied by both 'tea' and 'tee', but which is the solution is unclear, as least in the absence of other interlocking solutions to distinguish the final letter.)

3.2. Understanding a clue

Instances of the *SCML* are added to a clue as our understanding of its structure increases; it is an important part of our problem solving calculus to be able to capture increasing understanding, and many ways exist to do so. For this example we will simply write the addition of an annotation as a further entry in a table, as has been done in Table 1. There, beginning with an unannotated clue, we extend the table down-over as annotations are added.

A horizontal bar separates entries; the justification for a step is written to the right of the bar. Although of critical importance in an engineering setting, where justifications are typically used as the basis of an assurance argument, recording the justification accurately is an important element of any problem solving process. Here, without risk apparent, the justification is much less critical acting only as an *aide memoire*, or perhaps as a pedagogic tool.

⋮	'Installed' is an embedding indicator; 'shuddering' is an anagram indicator
'Note;(the)shuddering$_{a}$;(domestic appliance,(Bill)regularly$_{t}$)installed$_{e}$;noisy thing'	We have a complete explanation of the clue's structure!
	'HET' is an anagram of 'the'
'Note;HET;(domestic appliance,(Bill)regularly$_{t}$)installed$_{e}$;noisy thing'	'G' is a note (added 'A', then 'B', then ... to 'HET') until something fitted, 'GHET'
'G;HET;(domestic appliance,(Bill)regularly$_{t}$)installed$_{e}$;noisy thing'	Check: 'BL' are the alternate letters of 'Bill' and 'TOASTER' is a domestic appliance.
'G;HET;TO-BL-ASTER'	

Table 2: Extending problem understanding with solution exploration: 'Note the shuddering domestic appliance Bill regularly installed, noisy thing'

3.3. Moving towards solution

The process so far has dealt with the problem part of a cryptic clue: we have been exploring the problem. As the clue's structure becomes better understood, we are led to explore the solution. At any point in the development of a marked-up clue, we may have ideas about parts of the solution, for instance, resolving '(the)shuddering' to 'HET', and 'Note' to 'G' and fitting them together as pieces of a jigsaw puzzle. Such understanding of the solution also has a place in PO, and its recording made in the same way.

A possible path to solution leads us to continue Table 1 as we move towards increased solution understanding, adding justification as we go; see Table 2. Tables 1 and 2 together present, what might be called, a perfect development: problem understanding is completed before solution understanding begins. However, there is no a priori reason for solution understanding to have to wait for problem understanding in this way: in general, problem and solution understanding interleave in complex ways.

3.4. Doing it wrong

Of course, finding the correct structure from indicators is a difficult task and one that – in all probability – we would not get right first time. For instance, consider[7]:

Trajectory of missile soldier shot high in air over area

7. 3 down, The Guardian Cryptic Crossword No 25,044. Set by Brendan.

The correct structure[8] is

Trajectory of missile; soldier; (shot high in air)over; area

There are other interpretations, complete or partial, that may have been constructed as candidates during problem understanding. One such is

Trajectory of missile; soldier;((shot(high)in(air))over; area)

as well as many others. Often many candidates are held together in the mind, mixing and mingling to a richer synthesis that is correct. From incorrect to correct obtains through *backtracking*, i.e., at the point that something is known to be incorrect, retracing one's steps back to a previous state or to an unadorned clue so that other structures can be tried is possible and appropriate. The ability to backtrack appears an important part of problem solving.

4. Dynastic pasta starter is winding you up (8)

The task we set ourself in solving a crossword is to fill in its answer grid, i.e., to solve each of its clues and to fit the answers together. Initially, we focus on solving individual clues, the easy ones filling in some blanks, those shared letters providing additional clues for the more complex one. This defining characteristic of crosswords distinguishes them from a simple list of clues, enriching the solving experience; practically, even clues that are initially unsolvable are always worth taking a second look at when more of their letters are known. This amounts to later problem solving being assisted by earlier[9] problem solving.

We can go further. As each problem has both a problem- and a solution-part, a moment's thought reveals the three possibilities for overlap between two cryptic clues and their solutions:

- solution/solution overlap;
- clue/solution overlap;
- clue/clue overlap.

Shared letters between solutions is an example of the first of these, when solutions overlap. Clue/solution overlap is also popular within crosswords. As an example, consider The Guardian crossword Number 25,046:

21a The victims of 9? (12)

in which the '9' indicates a dependence on the solution of '9 across', *viz.*:

8. With 'soldier' giving 'para'; 'shot high in air' giving 'lob', 'over' reversing 'lob' to give 'bol'; and 'area' contracting to give 'a'.
9. Correct, we should add: an incorrect 'easy' solution is very unlikely to help in a later 'harder' one.

9a Hold over aspiring beneficiaries? (4,5)

Such interdependence is intended to produce a temporal ordering in the solving process: 9 across should have a correct solution for 21 across to have a complete clue. The process is complicated by the tricky setter that intertwines clues and their solutions for apparent circularity; Araucaria, in Cryptic crossword No 25,106, sets

21,7 King with a 25 suffers terminal hurt (6,6)

alongside

25 Something very pleasant caused endless trauma to 21 7 (5)

the endless trauma caused by such circularity is broken by a focus on the 'King' and the wordplay in 'suffered terminal hurt'.

The last case, in which clues themselves overlap is – in its most often used incantation – simple to deal with, as the following example from the same crossword shows:

7 See 21

Our discussion of crosswords, now complete, becomes the basis of our reflection on our problem solving calculus.

5. Think knight without gain? (7)

On tangling In another setting ([7]) we have identified problem-part/solution-part overlaps as creating tangled problems, a term intentionally suggestive of complexity, messiness and of the difficulty facing the solver of one such. That paper contains a much more detailed description of their form – complexity is added by the richer structure of an engineering problem: neither do we limit a tangle to two problems nor to a single overlap; environment and need enter the fray[10]; problems may tangle with themselves.

Even so, the simple case of crosswords provide good insight: we have seen how a tangling of problems can make later the solving process easier and enforce an order on problem solving. Thus tangled problems both assist and constrain the problem solver. In the engineering case, however, unlike the simpler crosswords, there is more subtlety at work, and interestingly, that subtlety is not because of their more complex structure. Consider the problem(s) of building a hole-in-the-wall cash machine: amongst many other needs, such a device must be simultaneously secure – so that money is protected – and accessible – so that money is available. Na
"ively, to decompose these problems so to be able to solve them independently, i.e., solving

10. If you will forgive the pun.

one with no knowledge of the other, produces a machine that is secure and a machine that is accessible: the most secure machine is that locked in a concrete bunker deep underground; the most accessible machine is an open one. No amount of glue, solder, or intellect can combine those solutions to produce a hole-in-the-wall. The difficulties we create in separating the problems are those of decomposing them to independent subproblems: such naïvety does not help the problem solver.

Rather, if we wish to untangle – and there are very good reasons to as experts tend to specialise around solution technologies – we must carry with us some record of the tangle, and use it to guide an expert's abilities in finding a solution.

For crosswords, the situation is simple; engineering is more complex.

On backtracking as problem and solution understanding With *SCML* we have provided a simple mechanism for recording understanding of the structure of a cryptic clue. As understanding develops two things can occur: either further markup is added; or the understanding is that the current structure isn't correct and we must backtrack to be able to continue. As in crosswords, the testing of candidate problems and solutions occurs as an integral part of engineering problem solving. However, in engineering, retaining an incorrect backtracked candidate for future use is as important as the ability to backtrack: backtracked developments are often as rich a source of understanding in a future development as are successful ones ([2]).

Another complication is that we may not be able to limit the backtracking that arises from the realisation of misunderstanding to that problem: in the case of a cryptic crossword, a pencilled in solution candidate being incorrect requires either a new solution to be found or a new clue structure candidate to be found. In the worst case, when dependence on a pencilled-in letter has lead to other solutions being incorrect, backtracking may require clues to be redone; such is the result of the tangling of problems. In engineering, the situation is worse: as tangling is not limited to a single level but – typically – permeates the whole engineering endeavour! Through a cascade of backtracking might arise a realisation that the problem is insoluble with the consequent loss of all problem solving resource invested!

Many, many modern engineering projects end this way, but then engineering is more complex than crossword solving.

On stakeholders' interests Unlike crosswords, engineering most often has a complex structure of stake-holders whose interest is in ensuring that their problems are solved[11]. The result is that both problem and solution understanding are more difficult – for reasons such as: the multiplicity of languages involved; conflicting and hidden stake-holder needs; difficulties in stake-holder communication; lack of problem solving resource; poor understanding of potential solutions' organising structures and sub-structures; the different media upon which solution parts are expressed; the needs of assurance; and there are many others.

11. This is a slight simplification. For instance, many stake-holders will not make their problems explicit, nor need they be aligned with those of other stake-holders or their organisation.

Again, for crosswords the situation is simpler than for engineering.

On rules Software Requirements Engineering brings with it, in an idea of Jackson [8], a way of leveraging the environment of the software system and the software requirements to make them both simpler. In our writings with Jackson we have developed a detailed implementation as a problem transformation called, following his suggestion, problem progression. It is interesting to note that, without the richer structure of an engineering problem, problem progression does not apply to cryptic clues but is specific to our problem solving calculus as applied to engineering, i.e., POE.

Compare this to the ruled-based capture of problem and solution understanding from Section 3.2 which can be seen as crossword instances of the problem interpretation rules of POE, rules, therefore, shared by both contexts. Problem interpretation (an application of which was surreptitiously given in separating the definition part in Table 1) and solution interpretation (which has not been used) are also at the level of our problem solving calculus.

This suggests a relationship between calculi which is not too unlike that existing between – at the base level – the propositional calculus and – above – its specialisations, for instance, to the predicate calculus [9]. In our problem solving calculus, therefore, built on the generic problem-part/solution-part problem model, there remain interpretation and expansion rules for both problem- and solution-parts. We expect also that the structuring mechanisms that exist to add detail in problem- and solution-parts, that we have termed *AStructs* and that generate co-design problems, must also be at the base level.

6. Italian with little conscience cut foreword to finish book (11)

We have extended the forms of problems that we can treat from the (very large class of) engineering problems to cryptic crosswords. From this extension, we have been able to distance the problem solving calculus underpinning our theory from its current expression on engineering problem forms.

In doing this, we have not been creative in our approach to clue solving, only in our ability to capture the problem solving process which leads to solved crosswords. To do this, we have simply encoded some small part of the techniques available to the recreational cryptic crossword puzzle solver from web-resources. And, whereas it would be interesting to compare our encoding with the algorithms that that resource provides, is it not critical to our extensions that we do so; it was not our intention to extend the techniques available to the solver.

Ostensibly, cryptic crosswords and engineering have little in common. However, both fit within the same problem solving framework, as we have shown here, which can, in future work, be detailed and presented as an abstraction above the problem form.

References

[1] Derek Mannering, Jon G. Hall, and Lucia Rapanotti. Safety process improvement with POSE & Alloy. In Francesca Saglietti and Norbert Oster, editors, *Computer Safety, Reliability and Security (SAFECOMP 2007)*, volume 4680 of *Lecture Notes in Computer Science*, pages 252–257, Nuremberg, Germany, September 2007. Springer-Verlag.

[2] Ann Nkwocha, Jon G. Hall, and Lucia Rapanotti. Design rationale capture in the globalised enterprise: An industrial study. In *Proceedings of Fifth International Conference on Software Engineering Advances (ICSEA 2010)*. IEEE, 2010. Electronic proceedings.

[3] Jerry Overton, Jon G. Hall, and Lucia Rapanotti. A problem-oriented theory of pattern-oriented analysis and design. In *Proceedings of IARIA PATTERNS 2009*, 2009.

[4] Lucia Rapanotti and Jon G. Hall. Designing an online part-time master of philosophy. In *Proceedings of the Fourth International Conference on Internet and Web Applications and Services*. IEEE Press, May 24-28 2009.

[5] G. F. C. Rogers. *The Nature of Engineering*. The Macmillan Press Ltd., 1983.

[6] Jon G. Hall and Lucia Rapanotti. Assurance-driven design in Problem Oriented Engineering. *International Journal On Advances in Systems and Measurements*, 2(1):119–130, 2009.

[7] Jon G. Hall. The global knot: How problems tangle in the world's economy. In Miltadis D. Lytras, Patricia Ordonez de Pablos, W.B. Lee, and Waldemar Karwowski, editors, *Electronic Globalized Business and Sustainable Development through IT Management: Strategies and Perspectives*. IGI-Global, 2010. ISBN: 9781615206230 DOI: 10.4018/978-1-61520-623-0.

[8] Michael Jackson. Problem analysis and structure. *Proceedings of NATO Summer School, Marktoberdorf*, 2000.

[9] S. C. Kleene. *Introduction to Metamathematics*. Van Nostrand, Princeton, NJ., 1964.