CHAPTER 3

A Context for the Heuristic Selection of *ZDC* Formulations

In Chapters 1 and 2 we have seen that there are often many ways in which a problem can be formulated as a constraint satisfaction problem. Furthermore, decisions made at this stage in the problem solving process are very important and can have a dramatic effect on the eventual cost of solving the problem. Examples of this include different *ZDC* formulations of the magic series problem and the Schur problem which showed order of magnitude differences in search cost.

There are many different aspects of constraint satisfaction research which affect the process of *ZDC* formulation selection. An example of this is the area of problem transformation techniques. Other areas include the investigation of the properties of CSPs and estimating the complexity of searching for solutions to a particular problem. These all have a role to play in the process of formulation selection. However, in the past, this role has not been made clear.

The aim of this chapter is to develop a context in which many aspects of CSP research can be related to *ZDC* formulation selection. In this way, we provide a focus for research in this area which, we believe, will result in improved techniques for achieving more reasoned selection. We demonstrate the importance of our new context which will be used in later chapters of this thesis.

3.1 The Space of Possible *ZDC* Formulations

In section 1.3.2 we described the different approaches to generating a set of candidate *ZDC* formulations of a problem. The starting point of this process is to translate the original problem description into an instantiation of *Z*, *D* and *C*. In fact we can generate many *ZDC* formulations in this way. Further formulations are then possible using transformation techniques.

Some transformations can result in very different *ZDC* formulations being created, while other transformations can result in *ZDC* formulations with only very subtle differences in their characteristics. Transformations, combined with manual generation can lead to a large spread of *ZDC* formulations, some having very similar properties and others having very different ones. The idea of there being such a spread is illustrated in figure 3.1.

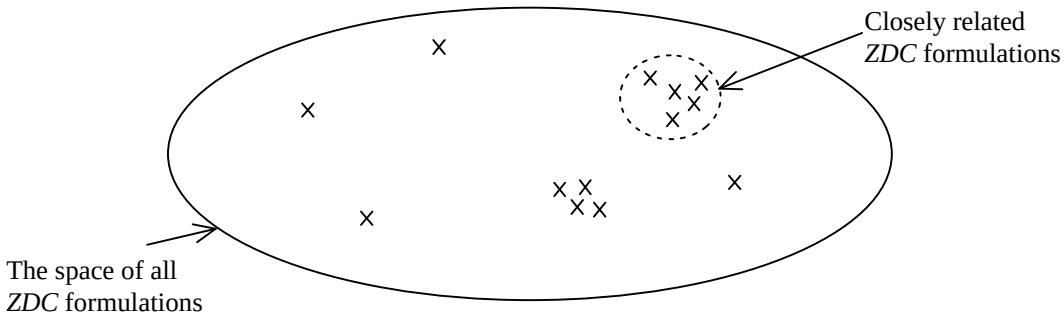

Figure 3.1 - The space of possible *ZDC* formulations for a given problem

One restriction which must hold within the *ZDC* formulation space for a given problem is that they must all be equivalent. By this we mean equivalent in terms of the definition used by (Rossi et al 1990) whereby the solution sets of different formulations are said to be *mutually reducible*.

Definition 3.1 (Rossi et al 1990) - Two CSPs P_1 and P_2 are *equivalent* if P_1 is reducible to P_2 and P_2 is reducible to P_1. This is written as $P_1 \equiv_e P_2$. ■

A CSP P_1 is reducible to CSP P_2 if there is a way to go from the solutions of P_2 to the solutions P_1 by mapping variables and values in P_2 into variables and values in P_1. When this mapping works in both directions the problems are said to be mutually reducible.

3.1.1 The SENDMORY Puzzle: an Example *ZDC* Formulation Space

As an example space of candidate *ZDC* formulations, we consider the crypto-arithmetic SENDMORY puzzle. This problem is shown in figure 3.2.

```
          S E N D
+         M O R E
=     M O N E Y
```

Figure 3.2 - The SENDMORY problem

The aim of the SENDMORY puzzle is to assign a unique digit to each of the letters used in the summation. One candidate *ZDC* formulation which we could have is *SENDMORY_1*;

SENDMORY_1:

Z: The eight letters of the problem

D: Each variable can be a digit and so has the domain {0,...,9}

C: C_1 all variables have different values

C_2 - $1000(S+M) + 100(E+O) + 10(N+R) + D + E = 10000M + 1000O + 100N + 10E + Y$

A second candidate *ZDC* formulation is possible if we use carry variables in order to split up the large equality constraint. This gives us *SENDMORY_2*;

SENDMORY_2:

Z: The eight letters of the problem plus three carry variables *a*, *b* and *c*

D: Each letter variable can be a digit and so has the domain {0,...,9}
each carry variable has the domain {0,1}

C: C_1 - all letter variables have different values

C_2 - $D + E = Y + 10a$

C_3 - $N + R + a = E + 10b$

C_4 - $E + O + b = N + 10c$

C_5 - $S + M + c = O + 10M$

We can see that our second *ZDC* formulation is significantly different from *SENDMORY_1*. It has more variables, and also larger search space complexity. The value of S is 10^8 for *SENDMORY_1* and $10^{8.9}$ for *SENDMORY_2*. Another feature of *SENDMORY_2* is that it has more diversity in its topology. For example, the constraints C_2 - C_5 have relatively low arities, ranging from 4 to 5. In contrast, the arities of the constraints C_1 and C_2 in *SENDMORY_1* are both 8.

As a third candidate *ZDC* formulation we can make some modifications to *SENDMORY_1*. More specifically, we can add three redundant constraints, C_3, C_4 and C_5, which provide additional explicit constraint based information about the nature of the solution set. This gives us *SENDMORY_3*;

SENDMORY_3:

Z: The eight letters of the problem

D: Each variable can be a digit and so has the domain {0,...,9}

C: C_1 - all variables have different values

C_2 - $1000(S+M) + 100(E+O) + 10(N+R) + D + E = 10000M + 1000O + 100N + 10E + Y$

C_3 - $S+M \leq 10M + O$

C_4 - $10S + E + 10M + O \leq 100M + 10O + N$

C_5 - $100S + 10E + N + 100M + 10O + R \leq 1000M + 100O + 10N + E$

SENDMORY_3 is very similar to *SENDMORY_1* since the only difference is the three redundant constraints C_3, C_4 and C_5. This is an example of two *ZDC* formulations that might be considered to be "close" in terms of the space of all *ZDC* formulations of the problem.

3.2 The Task of *ZDC* Formulation Selection

We have mentioned how current approaches to selecting the best *ZDC* formulation of a problem are somewhat ad-hoc in nature. This situation is undesirable since there are significant gains to be made from selecting the correct formulation and, conversely, there are significant losses that can be incurred by selecting a poor one. We should therefore like to take advantage of the potential gains, and avoid the potential losses. In this section we define the task of *ZDC* formulation

selection and the goals of an ideal *ZDC* formulation selection method. We follow this by a discussion of what we believe to be practical and achievable goals.

3.2.1 The Task Defined

We have seen how there are many ways problems can be formulated as CSPs. In addition, we notice that different algorithms are affected in different ways by different CSPs, and hence by different *ZDC* formulations. This was demonstrated in (Tsang et al 1995) and (Kwan 1997). These facts lead us to the definition of the primary task of the *ZDC* selection process;

Task 3.1: Let *cost(a, f)* be the cost of solving an individual *ZDC* formulation *f* using algorithm *a*. Given a problem *p* and the set of all *ZDC* formulations *F*, the task of the selection process is to find $f \in F$ such that we minimise the value of *cost(a, f)*.

This task assumes that we have committed ourselves to the use of a particular algorithm for solving the problem *p*. An extension of task 3.1 is to find the *ZDC* formulation which gives the minimum value of *cost(a, f)* for a set of algorithms, *A*. This leads us to a second task;

Task 3.2: Given a problem *p*, a set of *ZDC* formulations *F* and a set of algorithms *A*, the task of the selection process is to find $f \in F$ and $a \in A$ such that we minimise the value *cost(a, f)*.

The two tasks we have outlined above can be considered the requirements for an ideal *ZDC* formulation selection method. There are also some further requirements which are implicit in tasks 3.1 and 3.2. These are;

Generality - the method of selection should be general in that it can be applied to all classes of problem.

Accuracy - the method of selection should be highly accurate, ideally 100% accurate in choosing the most effective formulation.

Cost - the method of selection should have a low cost, ideally the cost should be nil.

Currently no such method exists and it seems unlikely that all three of these requirements can be satisfied simultaneously. However, a more reasonable proposition is that these ideal properties can act as a guide to the qualities of any practical selection method. This issue is discussed in the next section.

3.2.2 Expectations

Tasks 3.1 and 3.2 are extremely important tasks, as was seen, for example, with the magic number series problem in chapter 2. However, we must acknowledge that they are both impossible to achieve with certainty. The reason for this is that they depend on the availability of all possible *ZDC* formulations of a problem. Furthermore, the expression *cost(a, f)* cannot be 100% accurate unless we actually solve the *ZDC* formulation in question.

For any practical *ZDC* formulation selection method, we only ever have a subset of all possible *ZDC* formulations. For example, in section 3.1.1 we only considered three *ZDC* formulations of the SENDMORY problem. We also have to rely on approximations of *cost(a, f)*. For a practical method to be effective, we therefore rely on;

- a good pool of candidate *ZDC* formulations
- an effective method for approximating *cost(a, f)*

The higher the quality of these elements, the higher the quality of the selection and the closer we can get to fulfilling tasks 3.1 and 3.2.

Clearly, the only selection method which has zero cost is that of arbitrary selection. For other approaches, this cost should be taken into account. If this were not the case then we would have a simple solution to the 100% accuracy criterion whereby we solve each *ZDC* formulation of the problem before making our selection. Such an approach is unacceptable since it will always give worse overall search complexity than simply choosing one of the *ZDC* formulations at random and solving that one. We do not propose to put a figure on the cost that should be allowed in the selection process. However, it is desirable that this cost should be minimised and remain a low proportion of the overall solving cost.

3.2.3 The Approach in this Thesis

In this thesis we make a small but significant step towards achieving an effective and practical approximation to tasks 3.1 and 3.2. In this way we illustrate that it is possible to exploit the degree of freedom which exists when formulating a constraint satisfaction problem in order to reduce the cost of solving. As we shall see in later chapters, our contribution to the research community is particularly significant because it represents a step towards the possibility of automating the process of *ZDC* formulation selection.

Our approach to *ZDC* formulation selection is to develop heuristics based on the properties of CSPs. Many properties were described in chapter 2. One particularly useful property is that of the expected complexity of classes of CSP. We demonstrate how this can be used to develop effective heuristics for our selection method.

Throughout our work, we consider a set of search algorithms which consists of three very different systematic approaches. These algorithms are standard backtracking, forward checking (Haralick & Elliott 1980) and backjumping (Gaschnig 1979). This is a useful range of algorithms because it covers the important classes of lookahead and intelligent backtracking algorithms. The cost function used for these algorithms, *cost(a, f)*, is taken to be the measure of the number of constraint checks taken to complete the search.

We have said that the goal of 100% accuracy is unlikely to be achieved. The criterion for the success of our approach is based on a comparison with arbitrary selection. If we were to make arbitrary choices from a selection of n formulations then we can expect to select the most effective one $1/n$ times. As a result, the most basic criterion for an accurate formulation selection method is that we should select the most effective formulation more often than $1/n$ times. For a pairwise selection, this means 50% accuracy is the minimum requirement. Of course we should hope that our method for selection would produce an accuracy of selection better than this.

There are many elements of constraint satisfaction research that contribute to the process of *ZDC* formulation selection. Before we proceed with the details of our particular selection method, we develop a context in which these elements can be placed. Such a context is important in establishing the relationship between the different aspects of the selection process.

3.3 A Context for Selection

The aim of this section is to define a context for the selection of *ZDC* formulations of a given problem. Such a context has not been defined previously and as a result, many aspects of constraint satisfaction research which affect the *ZDC* formulation selection process remain unconnected. We believe our context will help to bring these together. It also acts as a focus for our work, allowing us to target specific aspects of the selection process. Furthermore, our context provides a useful framework which can act as a base model for automatic *ZDC* formulation generation techniques.

A problem formulation generator G takes a problem specification and generates an instantiation of *Z*, *D* and *C*. If $\mathbb{P}$ is the set of problem specifications within the domain of G and $\mathbb{F}$ is the set of all possible *ZDC* formulations then we have the mapping;

$$G: \mathbb{P} \rightarrow \mathbb{F} \qquad (3\text{-}1)$$

As we discussed earlier in this chapter, our view of *ZDC* formulation selection is one of a heuristic traversal of the space of all possible formulations, $\mathbb{F}$. If we are given a particular *ZDC* formulation f as a starting point, we can always move to alternative points in that space using transformation algorithms, or *move operators*. We then need two types of heuristic in order to facilitate that movement. The first type of heuristic is one that suggests an appropriate transformation that can be expected to produce an improvement on the current *ZDC* formulation. The second type is used to evaluate the actual result of any suggested transformation. These three elements of our context, move operators, suggestion heuristics and evaluation heuristics, are described in the following sections.

3.3.1 Move Operators and Transformation Algorithms

The role of move operators is to provide us with mobility through the space of possible *ZDC* formulations. There are two basic approaches to moving through this space. The first is to use that actual problem solver to generate alternative *ZDC* formulations. A second approach is to use *ZDC* formulation *transformation algorithms*;

Definition 3.2 - Given a problem *p* and a *ZDC* formulation *f*, a *ZDC* formulation *transformation algorithm T(f)* generates as its output an equivalent *ZDC* formulation for that problem. ■

This gives us;

$$T: \mathbb{F} \rightarrow \mathbb{F} \qquad (3\text{-}2)$$

An example *ZDC* transformation algorithm is the dual transformation (Dechter & Pearl 1989). The output *ZDC* formulation of this transformation, *f2*, is dual in the sense that for every constraint in the original, *f1*, we create a variable in *f2*. The domain of these variables is then determined to be the set of legal compound labels defined in the associated original constraint. This is seen in figure 3.3 where we have two variables in *f2*, corresponding to the two constraints C_{AB} and C_{AC} in *f1*. In order to ensure equivalence in both *ZDC* formulations, the final step in the transformation is to add constraints between variables in *f2* which originate from constraints in *f1* having common variables. In our example, x_{AB} and x_{AC} have a common *f1*-variable, namely x_A. The constraint $C_{AB,AC}$ simply ensures that the identical label is used for the common *f1* variable.

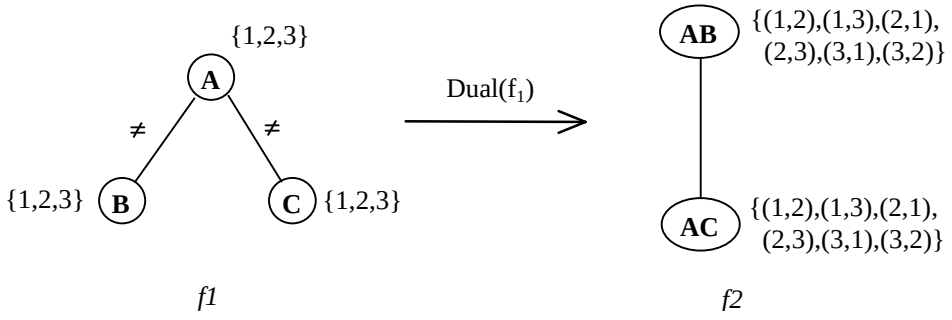

Figure 3.3 - The dual transformation

An interesting property of the dual transformation is that it always produces a binary CSP as its output.

Many other transformations are possible. Examples include;

- problem reduction techniques (Tsang 1993) such as arc-consistency (Mackworth 1977)
- abstraction (Schrag & Miranker 1996) (Freuder et al 1995) (Freuder & Sabin 1997)
- the addition and removal of redundant constraints (VanHentenryck 1989) (Dincbas et al 1988) (Smith 1996) (Borrett 1998)
- the merging of variables (Dechter & Dechter 1987)

Having such a range of transformations available means that we have more freedom to move within the space of all possible *ZDC* formulations. We may also use combinations or sequences of transformations in order to generate yet more *ZDC* formulations. For example, we could use the dual transformation followed by the removal of certain redundant constraints.

3.3.2 Suggestion Heuristics

As the number of *ZDC* transformation algorithms available to us increases we gain an increasing level of mobility within the space of *ZDC* formulations. However, this gain does not come without cost and we only want to apply transformations that are likely to produce beneficial effects. Simply using all transformation algorithms available to us is not a sensible path to *ZDC* formulation generation. What we should like is to use heuristics which guide us to sensible moves through the *ZDC* formulation space. This is the role of *suggestion heuristics*.

Definition 3.3 - Given a *ZDC* formulation *f* and a suggestion criterion *g*, the role of a *suggestion heuristic*, H_s, is to select one or more transformation algorithm, T, which is expected to improve the suggestion criterion *g*. ■

This gives us;

$$H_s: \mathbb{F} \times \mathbb{G} \rightarrow \mathbb{T} \qquad (3\text{-}3)$$

where $\mathbb{T}$ is the set of available *ZDC* formulation transformation functions, where $\mathbb{G}$ is the set of suggestion criteria.

The key to an effective suggestion heuristic is the choice of the suggestion criterion, *g*. Sensible candidates are criteria based on the properties we described in chapter 2. For example, the search space complexity, *S*, is a potentially useful candidate. A suggestion heuristic based on this criterion might point us towards the use of problem reduction algorithms or the merging of variables, both of which fulfil the criterion. It would not recommend transformation algorithms which can are likely to increase *S*, which is sometimes the case with the dual transformation.

One view of suggestion heuristics is to compare them with variable ordering heuristics. These are usually chosen before search begins. They can be extremely effective, but no variable ordering heuristic has yet been found which is the universal champion, to be used for all problem classes. Results to this effect were seen in (Tsang et al 1995). In similar way, we believe *ZDC* formulation suggestion heuristics may have domains according to the particular *ZDC* formulation they are operating on.

3.3.3 Evaluation Heuristics

The most important element of tasks 3.1 and 3.2 is the mechanism for evaluating the function *cost(a, f)*. Without such a function, we are left with making an arbitrary choice between *ZDC* formulations. Since we do not know of any ideal selection method, our objective is to obtain approximations to the cost function. In other words we need to find heuristics which allow us to make good decisions. We call such heuristics *evaluation heuristics*.

Definition 3.3 - Given two *ZDC* formulations *f1* and *f2*, an *evaluation heuristic*, H_e, returns a tri-state value which gives an indication of the relative expected cost of solving them. A value of *1* denotes that *f1* is expected to have the cheapest solving cost, a value of *-1* denotes that *f2* is expected to have the cheapest solving cost and a value of *0* denotes that there is no expected difference. ■

This gives us;

$$H_e: \mathbb{F} \times \mathbb{F} \rightarrow \mathbb{C} \qquad (3\text{-}4)$$

where

$$\mathbb{C} = \{ 1, 0, -1 \}$$

In chapter 2, we described a selection of properties of CSPs. Many of these can be used to give an indication of the expected problem solving cost. Since it is not feasible to actually solve candidate *ZDC* formulations before making any selection, H_e must be based on a subset of such properties, or measures. In other words, we have a vector of measures, M, which is used by our evaluation heuristics;

$$M = (m_1, m_2, \ldots, m_k)$$

where each element of the vector can be expressed as;

$$m_i: \mathbb{F} \times \mathbb{F} \times \mathbb{A} \rightarrow \mathbb{C} \qquad (3\text{-}5)$$

or

$$m_i: \mathbb{F} \times \mathbb{F} \times \mathbb{A} \rightarrow \mathbb{R} \qquad (3\text{-}6)$$

Where $\mathbb{R}$ represents the domain of real numbers, and $\mathbb{C}$ is used to denote the relative merits of a certain property.

Using this multi-measure approach does introduce a complication, however. For example, consider the two *ZDC* formulations given in figure 3.3. Supposing we had a vector including the two measures search space complexity, *S* and average degree, $\overline{d}$. As we mentioned in chapter 2, in general, it might be considered desirable to have a lower value of *S* since it restricts the worst case performance of the any algorithm. Similarly, a lower average degree might be considered a desirable feature, all other properties being equal, since it reduces the number of constraint checks an algorithm needs to perform at each node in the search space. Assuming this to be the case, we would get conflicting information about which *ZDC* formulation to choose;

- property *S* suggests *f1* to be the preferable to *f2*, based on values of 27 and 36 respectively
- property $\overline{d}$ suggests *f2* to be preferable to *f1*, based on values of 1 and 1.33 respectively

Clearly this is only a very small and simplified example, but it serves to demonstrate how conflicting information given by different measures in the vector M must be resolved. It is the role of evaluation heuristics to perform this resolution function and, as a result, their design is directly related to the choice of measures upon which they are to be based.

Since there is an almost unlimited source of potential measures that can be defined, the key to effective evaluation heuristic design is the use of properties which provide a good correlation between their value and the cost of solving the *ZDC* formulations in question. Furthermore, while increasing the number of properties used in M may provide a greater resolution of discrimination between *ZDC* formulations, there is a trade-off between the length of M and the complexity of the resolution function.

3.4 Discussion

We have described a context for the heuristic selection of *ZDC* formulations which allows us to focus on areas of constraint satisfaction research which affect the *ZDC* formulation selection process, both directly and indirectly. This three main elements of our context are;

- move operators
- suggestion heuristics
- evaluation heuristics

Of these, there is a significant amount of work which has already been done in relation to *ZDC* formulation transformation algorithms, which act as move operators. However, we believe that relating such algorithms directly to our context will allow us to view them more directly in association with the *ZDC* formulation selection process. This contrasts with the traditional view of simply being a pre-processing step carried out before search, as is often the case with problem reduction techniques such as arc-consistency.

The main focus of the remainder of this thesis is to develop the largely neglected area of *ZDC* formulation evaluation heuristics. We mentioned in section 3.3 that the key to this area was the ability to resolve a vector of measures, M. One possible approach to this is to adopt the use of multi-variate analysis techniques (Tabachnick & Fidell 1996). These could be combined with empirical data obtained from solving different *ZDC* formulations of a wide range of problem classes.

Another candidate for the design of effective evaluation heuristics is to use a hierarchical approach. This works by applying measures in their order of significance and then making decisions based on that order. As an example[1], suppose we have a measure vector M which consisted of three properties, *T-Factor*, *S* and *p1*. A possible hierarchical resolution function based on these properties is seen in figure 3.4.

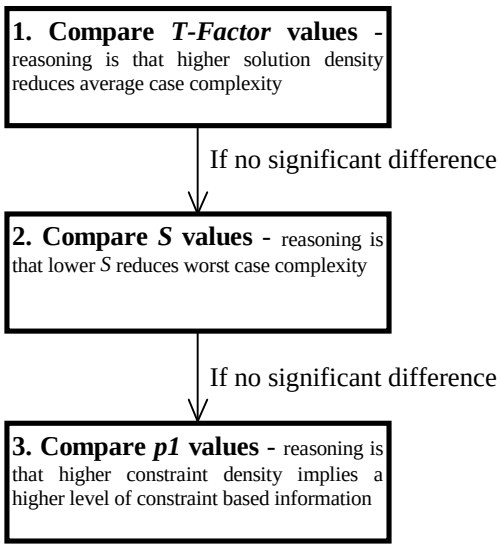

Figure 3.4 - A possible hierarchical vector resolution algorithm

Remembering that we are interested in developing heuristics, both the multivariate analysis approach and the hierarchical approach present possible ways forward. This is a new and important area of research and as a result there are many unexplored avenues.

[1] Note that this example is for illustrative purposes only. We do not have evidence to support its validity.

One promising line that we have developed is the use of measures of theoretical complexity as a useful measure in the vector M. This property was discussed in chapter 2 and as we shall show in subsequent chapters, it can be used as the dominant element of M for many classes of problem. As a result, it simplifies the process of measure vector resolution which evaluation heuristics need to perform.

3.5 Summary

Our main contribution in this chapter has been to outline a context for the heuristic selection of *ZDC* formulations of a given problem. This is a new and important development because it clearly identifies the features of constraint research which affect the *ZDC* formulation selection process. It also allows us to focus on the areas of research which directly affect the selection process. Furthermore, the context we have outlined provides us with a useful framework in which automatic *ZDC* formulation generation techniques can be developed.