Exploring Feasible Optimisation Technique of Packing Measurands in Data Cycle Map

Hiren Mistry
Department of Engineering Science
The University of Auckland
Auckland, New Zealand
hmis001@ec.auckland.ac.nz

Abstract

During flight test, a common objective is to collect data from specially instrumented aircraft. Typically several hundreds of measurands (parameters) such as speed, altitude, and pressure, are monitored and telemetered to the ground receiving station for both real-time monitoring and storage for later post-test analysis. Prior to this transmission, measurands are digitally represented and placed in a certain order within a data structure know as Data Cycle Map (DCM). These placements are based on measurands periodicity, sample rate and certain industry standards. Therefore, it is necessary to determine the order of placement before hand.

The process of designing data cycle map can be performed manually for few measurands but as the number of measurands increases the process becomes rather complex and time consuming. The need to improve this process has become an interesting topic to explore in operation research group.

In the past the problem of designing DCM has been modelled using set covering integer-programming formulation. In this approach measurands (the parameters) are allocated to the map and the DCM design is optimised in terms of the amount of wasted space on the map. This problem is equivalent to Multi Choice Knapsack Problem (MCKP). As number of measurands increases, the solve time increases exponentially. It is therefore *NP*-hard and clever optimisation technique is needs to solve these problems. The focus of this paper is to investigate a new mathematical approach to strengthen the set-covering model described in the past. It is desired that solution based on branch and bound process can be speed up by generating strengthened LP solution.

1 Introduction

Data Cycle Map (DCM) is a telemetry system used to transfer data from one site (vehicle) to another (ground monitor or controller). This telemetry is required for the transmission of data and video necessary for the safe and efficient conduct of flight test and training missions. Many programs require telemetry during high-risk phases of air flight testing (Aerodynamic Flutter Analysis, Structural Loads, Performance and Flying Qualities, Engine Integration, etc.) for the safe conduct of the mission to prevent aircraft damage or loss. On some flight missions, due to their non-recovery nature, telemetry is the only data source. There ground facility staff uses real-time data to monitor, control, and integrate training exercising. For accelerated flight test programs where real-time data determines the next mission, the requirements for real time data push the limits of existing telemetry capabilities. Coupling this with, the shrinking spectrum allocated for telemetry, results in a serious problem for the flight test and training mission.

Since telemetry system is also use for land vehicles testing and sea born crafts (ships, submarines), we required different telemetry system depending on the nature of tests. It is not feasible to have one telemetry system to fit all test. Therefore we need to design the telemetry system (like the data cycle for flight test) before hand, depending on the nature of test. The process of designing this system is rather complex and time consuming, because of the need to place data periodically in the data cycle map frame. To build up on the problem of designing data cycle map, the section below describes the basics of telemetry, followed by the structure of Data Cycle Map required for flight test.

1.1 Data Cycle Map (DCM) Basics

DCM is a telemetry system used to acquire data measurands (like speed, altitude, pressure etc.) in one location and encode them for transmission. The Figure 1 shows the basic elements of a DCM telemetry system. A DCM encoder converts the input data signals into a serial data format suitable for transmission. At the receiving end, a DCM decoder converts the serial data back into individual output data signals (Herley-Metraplex, 1998)

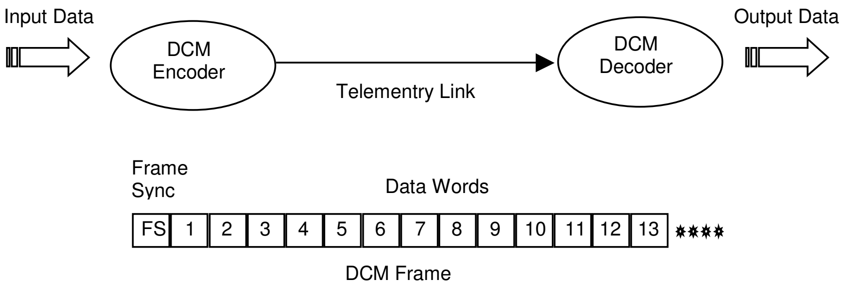


Figure 1: Data Cycle Map Basics

The data is transmitted as a serial stream of digital words consisting of logic ones and zeros. When making large numbers of measurements, it is desirable to squeeze the data into one signal or data link in order to simplify the transmission and recording.

The simplest DCM frame consists of a frame synchronization word followed by a string of data words. The frame repeats continually to provide new data samples as the input data changes. Frame synchronization enables the DCM decoder to easily locate the start of each frame.

1.2 Data Cycle Map Structure

The DCM data structure is an array made of major frame whose rows are called minor frames. Measurands are allocated onto blocks within this data structure. The placements is based on measurands periodicity, sample rate and certain standards set by Inter Range Instrument Group (IRIG) (IRIG Telemetry Group, May 1996). Since these standards are too many to discuss, we summarise some of the standards, which will impact the DCM design:

- A major frame contains no more than 256 minor frames.
- The length of minor frames cannot be more than 512 16-bit words in length.
- Each measurand must be placed periodically on the map with accordance to their sample rates and word length.
- Each major and minor frame must start with frame synchronisation words and contain a frame ID.
- Data Cycle Map is a repetition of several major frames

From the above standards, the construction of data cycle map would be effected mostly by the need to place measurands periodically on the major frame in accordance with their sample rate and word length. Making the DCM construction process complex and time consuming.

One can generate the data cycle map manually for small data sets with few measurands but with several hundreds of measurands to be placed on the map, manual processes would be faced with coincident placements. There is a greater chance of having empty spaces in the data cycle map. It is necessary that the process of data cycle map construction is efficient as possible. By efficiency we mean that the empty spaces are reduce and the solution is derived in minimal computation time.

We can apply optimisation methods to resolve this problem but our ability to manage this task in terms of computational memory should be taken into account. The construction of entire major frame would be bounded by computational limitations. To take this into account we reduce the problem into smaller tasks whose solutions can be used as building blocks for solving the larger problem (P. David, C. Stephen, 2001).

Our approach would be to take the measurand periodicity and sample rates in account and construct minor frames, which could then be replicated to develop the entire major frame. To apply our approach we would represented each measurand in the DCM by specified number of words. Each word will occupy a 'slot' in the DCM. The approach will be described with an example in the next section along with the structure of minor frame and the computation of minor frame length.

2 Structure of minor frame

To be consistent with the earlier discussion, we consider the construction of a single minor frame whose replication will be create to generate the entire major frame. The structure of minor frame is dependent on several factors. Apart from periodicity and minimum required sample rates, the IRIG standards mentioned earlier has to be taken into account.

It is possible that some parameters are collected at lower sample rates that others, such measurands may not appear on every minor frame, we call them subcommutated. On the other hand, measurands, which occur at least once on each minor frame, are said to be supercommmutated. Lets look at the insight into the DCM construction process by working through a simple example.

Consider the following notation

n	=	The number of measurands
s_i	=	The major frame sample rate for measurand i
m_i	=	The minor frame sample rate for measurand i
p_i	=	The placements period in the minor frame for measurand i

b_i	=	The required number of bits for measurand i
lsr	=	The lowest sample rate of all measurands in the DCM
mfl	=	The minor frame length.

Example 1: The eight measurands data is shown in Table 1. For the purpose of this example we assume that global word length is 16 bits (i.e. each slot in the minor frame is 16 bits.). This may vary from one data set to another. If the required number of bits for the measurand is 24 then two slots will be used in the minor frame. Else if, the required is 32 bits then 4 slots will be used in the minor frame and so on. It is assumed that the header words are placed at the beginning of each minor frame, but the frame id may be placed anywhere in the frame. Within these limitations our aim is to construct minor frame, which satisfies measurand periodicity and sample rate.

Measurand Number	Sample Rate, s_i	Measurand Period, p_i	Nos of Bits, b_i	Minor Frame Sample Rate, m_i
1	36	6	16	3
2	24	9	16	2
3	24	9	32	2
4	24	9	16	2
5	12	18	16	1
6	12	18	16	1
7	12	18	32	1
8	12	18	48	1

Table 1: Eight Measurand Problem

To ensure the placement of measurands in each minor frame we select lsr, the minor frame length as the lowest sample rate of s_i (i.e. 12). This is used to normalise the sample rates of all measurands by constructing minor frame sample rates m_i given by s_i / lsr. The value of m_i represents the number of times a measurand would appear in a single minor frame. In example 1, measurand one would appear 3 times, measurand two 2 times and measurand five 1 time.

Having determined the sample rate in minor frame for each measurand, the main difficulty lies in satisfying the periodic requirement. This is essential because once the minor frame is constructed; the DCM is obtained by repeating the minor frame lsr times. Given the values of m_i and b_i, the minor frame length mfl is given by $\sum_i m_i(b_i/16)$. For example 1, the minor frame length is 18. It is important to realise that mfl is dividable by each p_i. Therefore periodicity of the measurand is achieved. The figure 2 shows one possible construction of minor frame for example 1.

Slots	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Measurand Placement	1	2	4	6	3	3	1	8	8	8	2	4	1	3	3	5	7	7

Figure 2: Single minor frame construction for example 1

It shows that to maintain periodicity, if measurand four is placed in slot 1 then the next placement would be in slot 7 and 13. Similarly, if measurand four is placed in slot

3 then the next placement would be in slot 12. This proved that the measurands periodic nature is achieved.

Now, having determined the length and the structure of minor frame, the complete major frame is then constructed by replicating the minor frame as many times as the lowest sample rate lsr, which is 12 times in this case. Thus solving the problem of placing measurand in the minor frame.

In example 1, the periodicity of measurands is satisfied because all p_i divides the minor frame length mfl. However, it is not always guarantied that mfl will be divided by all p_i's. This can be proved when the required sample rate for measurand four is changed to 36. It would now change the minor frame sample rate from 2 to 3. Making the minor frame length 19, which cannot be divided by all p_i. So it doesn't matter were the measurands 1,2,3 and 6 are placed on the minor frame. It will not satisfy the periodicity of measurands with mfl 19. This behaviour will be explained in section 2.1.

It is therefore necessary to allow some allowance to achieve periodicity. When the minor frame length is not divisible by p_i's we choose mfl to be the least integer greater than the minor frame length. This value is a multiple of the lowest common multiple (LCM) of all p_i values and will ensure the divisibility of mfl / p_i. For the scenario discussed earlier, the minor frame length is not divisible by all p_i. Therefore we choose the value of the mfl to be the lowest common multiple of all minor frame sample rates p_i (i.e. 20). This will ensure the periodic placement of measurands.

However this doesn't necessary give a "feasible placement" of measurands in the minor frame. By "feasible placement" we mean that measurands are placed in the frame satisfying the non-coincidence placements and the periodic constraints. It can be proved that by increasing the LCM by certain factor coincidence placement can be avoided.

For the context of this paper, research has been involved in finding the "feasible placement" of measurands in the minor frame. In section 3, we construct a 0-1 integer formulation to solve this problem. Two models have been discussed and their ability to generate "feasible placement" is compared.

2.1 Why coincidence placement occurs?

Lets explain the coincidence placement with the help of an example. Suppose, if we have two measurands with sample rates 3 and 7 respectively with required bits 16. The frame length is computed by taking the LCM of 3 and 7, which is 21. With measurands periods of 3 and 7 it can be shown that, a non-coincidence placement is impossible. It doesn't matter were the two measurands are placed in the frame the coincidence will always occur. This situation arises in the above example because of the relative prime relationship between the two periods 3 and 7. This can be proved with theorem 1.

Theorem 1: Let $m_1, m_2 \ldots.. m_n$ be the sample rates of n measurands in a minor frame with associated periods $p_1, p_2 \ldots.. p_n$. A sufficient condition for the coincidence of parameters i and j in the frame is that periods m_i and m_j are relatively prime (P. David, C. Stephen, 2001)..

Proof: Consider m_i and m_j, and their associated periods p_i and p_j, which are relatively prime. Assume that parameters i is first placed at position a_i, then subsequent placements are given by $a_i + s_i\ p_i$ with $0 \leq s_i < m_i$. Parameters j can now be placed at positions $a_j + s_j\ p_j$ where $0 \leq s_j < p_j$ and $a_j \neq a_i + k\, p_i$ for integer k. Now, consider the situation for which there is at least one value for which positions of parameters i and j coincide, namely $a_i + s_i\ p_i = a_j + s_j\ p_j$. This can be rewritten as $s_i\ p_i - s_j\ p_j = a_j - a_i$. This represents a diophantine equations in variables s_i and s_j. It is know that solutions for such equations is the greatest common divisor of p_i and p_j which divides $a_j - a_i$, i.e. $(p_i, p_j) \mid a_j - a_i$. We note that if p_i and p_j are relative prime then $(p_i, p_j) = 1$ which will always divide $a_j - a_i$. This means that whatever choice of initial values, a coincidence of positions will occur and hence periodic assignment is not possible in this case.

2.2 How to avoid coincidence placement?

In the above section we proved that the coincidence place is unavoidable when the periods of measurands have prime relationship. To overcome this situation, the length of minor frame is increase *x* times until the prime relationship is overcomed. For earlier example, with period 3 and 7 and minor frame length 21, coincidence can be avoidable by increasing the minor frame length by 2 times (i.e. 42).

3 Problem Formulation

This section presents an integer formulation of the Multi Choice Knapsack problem (MCKP), which has been extensively studied in the operations research community (S. Martello and P. Toth, 1990). It describes how the MCKP formulation can be used to solve the problem of packing measurands into a minor frame once the minor frame length is known. It is trivial proof that finding an optimal solution to the MCKP is NP-hard because it requires complete enumeration of solution space. Therefore, for the context of this paper, interest is plainly based in generating "feasible placement" of measurands in the minor frame while satisfying sample rate and periodicity. We describe two mathematical models - I and II, which would ensure this. These models will be investigated on their efficiency for generating feasible placements and their results will be discussed in section 4.

3.1 The Multi Choice Knapsack Problem

An interesting generalisation of the knapsack problem is obtained, if the set of variables *N* is partitioned into *m* classes N_k, $k = 1,2\ldots,m$ and if we require that exactly one variable has to be chosen from each class, we get the multi-choice knapsack problem (MCKP). It can be defined as:

Maximise: $$\sum_{k=1}^{m} \sum_{j \in N_k} c_{ij} x_{ij}$$

Subject to: $$\sum_{k=1}^{m} \sum_{j \in N_k} a_{ij} x_{ij} \le b \qquad (1.1)$$

$$\sum_{j \in N_k} x_{ij} = 1 \qquad i = 1,2,..m \qquad (1.2)$$

$$x_{ij} \in \{0,1\} \qquad i = 1,2,..m, \; j \in N_k$$

Here the binary variable $x_{ij} = 1$ states that item *j* was chosen from class *k*. The constraint 1.1 ensures that exactly one item is chosen from each class.

It is possible to solve the above problem exactly but as the number of classes increased, the solve time to find the optimal solution increases. Since the solution time is based on branch and bound enumeration, we do not have any other worst-case time bound than a complete enumeration in exponential time. The problem is NP-hard and requires a good initial LP solution. The aim of the two models described below is to generate good initial LP solution by relaxing the integrality constraint. So the time taken to generate integer feasible solution is reduce.

3.2 Measurand Placement Formulation (Model I)

The measurand placement problem is formulated using the MCKP defined earlier. Having determined the length of the frame (i.e the number of slots required to satisfy periodicity of measurands). The problem is then to place as many measurands as possible in the frame to minimise the unused space. The measurand placement problem is defined as follows:

$$\text{Maximise:} \quad \sum_{n=1}^{m} \sum_{j \in P_n} c_{ij} x_{ij} \tag{2.1}$$

$$\text{Subject to:} \quad \sum_{n=1}^{m} \sum_{j \in P_n} a_{ij} x_{ij} \le b \tag{2.2}$$

$$\sum_{j \in P_n} x_{ij} = 1 \qquad i = 1,2,..m \tag{2.3}$$

$$x_{ij} \in \{0,1\} \qquad i = 1,2,..m, \; j \in P_n \tag{2.4}$$

Where: m = number of measurands to be placed in the frame

P_n = sample rate for measurand n

c_{ij} = cost of selecting column j from measurand i

a_{ij} = weight for placing measurands in the frame

b = bound on the placement of measurand in a slot.

$$x_{ij} = \begin{cases} 1, \text{if column } j \text{ is selected from measurand } i \\ 0, \text{otherwise.} \end{cases}$$

Constraints 2.2 and 2.3 ensure that one column is selected from each measurand set so that the frame is covered with no overlap. Generation of a_{ij} automatically ensures that periodicity of measurands is maintained. To understand the formulation, consider the example described below.

Example 2:

Measurands	3

Nos of Samples	Samples per second	Nos of Bits
1	14	16
1	10	16
1	20	16

Length of Frame	LCM(14,10,20)	140

In this example, there are three measurands with sample rates 14, 10, and 20. Each measurands requires 16 bits of storage space. So we assume that each slot on the minor frame is 16 bits. To satisfy the periodicity constraint, the minimum frame is computed by taking the LCM of sample rates. In this case the LCM is 140 slots.

Having determined the length of the frame we can now enumerate all possible placements for a given measurand on this frame. Each placement is refered as a tour. The figure 2 below shows the tours generated for each measurands. The tours represent the structure of the A matrix for our knapsack formulation. The gub constraints 2.3 ensure that at least one tour is selected from each measurand set of tours.

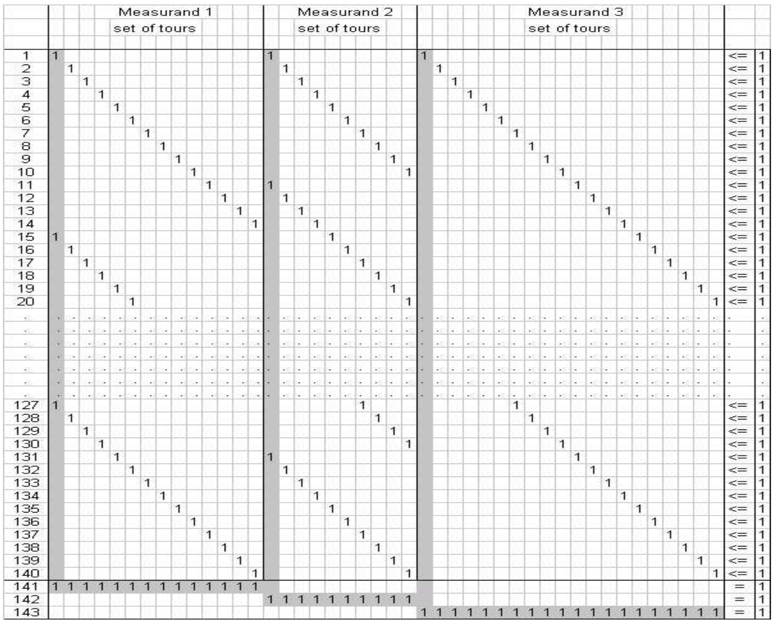


Figure 3: Sample A matrix for measurand placement formulation

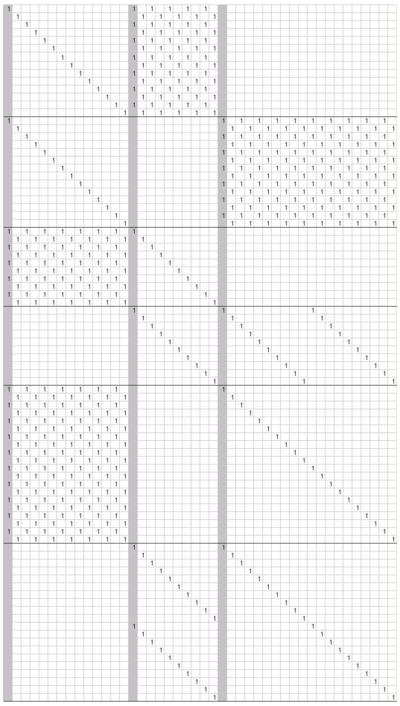
The solution derived from this model will give the starting slots for each measurands. But as the number of measurands increases the amount of time taken to generator integer solution from initial LP solution increases. It is therefore important to explore methods that would give lower LP, IP gap. So the time taken by branch and bound to compute the integer solution is reduced. To achieve this, we derived a second model, which is believed to be a strengthened model then I.

3.3 Measurand Placement Formulation (Model II)

This model is also based on MCKP formulation. In this approach we look at the compressed view for each measurands. By compressed view, we mean - how one measurand would view in period of another? To understand this, lets look at example 2 mentioned earlier. Here, measurand one and two has period 14 and 10 respectively. To generate compressed view of measurand two in measurand one, we compress the 140-slot frame mentioned earlier into 14 slots (i.e. the period of measurand one).

The figure 4 shows the compressed view. Each column represents a tour based on initial placement. For measurand 2, if the initial placement is in slot 1 then based on the measurand periodicity, the next placement would be in slot 11, 21, 31 and so on. Until the limit of minor frame length is reached (i.e.140). But since we are generating a compressed view of measurand two in one. The equivalent slot number is given by taking the modulus of the slot number divide by the size of compressed frame (i.e. 14). For example, if the placement of measurand is in slot 21 then the equivalent slot number for the compressed view is given by the reminder of 21/14 (i.e.7 in this case.). This gives the tours for measurands and thus the compressed view representation.

In logical thinking, compressed view represents the coincidence placements of the two measurands. The row 1 in figure 4 states that, if the initial placements of measurand one is in slot 1 then this will coincide with placement of measurands two in slots 1,3,5,7 and 9. Thus giving the overlapping constraints for measurand 1 and 2 in slot 1.



		Measurand 1														Measurand 2									
		period														period									
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	1	2	3	4	5	6	7	8	9	10
	1	1														1		1		1		1		1	
	2		1														1		1		1		1		1
	3			1												1		1		1		1		1	
	4				1												1		1		1		1		1
	5					1										1		1		1		1		1	
	6						1										1		1		1		1		1
Slots	7							1								1		1		1		1		1	
	8								1								1		1		1		1		1
	9									1						1		1		1		1		1	
	10										1						1		1		1		1		1
	11											1				1		1		1		1		1	
	12												1				1		1		1		1		1
	13													1		1		1		1		1		1	
	14														1		1		1		1		1		1

Figure 4: Compressed view of measurand 2 in measurand 1

Taking this concept of compressed view into account. The A matrix can be easily formulated as shown on the left in figure 4. The number of compressed view in the matrix is given by $^{n}C_{2}$, were n is the numbers of measurands. Once the A matrix is defined we can now apply reduction technique to reduce the size of A matrix.

One common reduction would be to add and lift constraints from the compressed view. For measurand 1-2 and 2-1, figure 5 summarises this technique. It shoes that constraint 15 can be derived by adding constraints 1,3,5,7,9,11 and 13. Giving constraint (1). The coefficients are then reduced by dividing the constraint by 7 and rounding the coefficients upwards to the nearest integer. But check needs to be made that the constraint derived has the same meaning as constraints 1,3,5,7,9, and 11. Giving the constraints (2). This new constraint also dominates constraints 17,19,21, and 23. Therefore we eliminate these constraints and replaced it with new constraint (2).

			Measurand 1														Measurand 2											
			period														period											
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	1	2	3	4	5	6	7	8	9	10		
Constraints	Pairwise view 1-2	1	1														1		1		1		1		1		<=	1
		2		1														1		1		1		1		1	<=	1
		3			1												1		1		1		1		1		<=	1
		4				1												1		1		1		1		1	<=	1
		5					1										1		1		1		1		1		<=	1
		6						1										1		1		1		1		1	<=	1
		7							1								1		1		1		1		1		<=	1
		8								1								1		1		1		1		1	<=	1
		9									1						1		1		1		1		1		<=	1
		10										1						1		1		1		1		1	<=	1
		11											1				1		1		1		1		1		<=	1
		12												1				1		1		1		1		1	<=	1
		13													1		1		1		1		1		1		<=	1
		14														1		1		1		1		1		1	<=	1
	Pairwise view 2-1	15	1		1		1		1		1		1		1		1										<=	1
		16		1		1		1		1		1		1		1		1									<=	1
		17	1		1		1		1		1		1		1				1								<=	1
		18		1		1		1		1		1		1		1				1							<=	1
		19	1		1		1		1		1		1		1						1						<=	1
		20		1		1		1		1		1		1		1						1					<=	1
		21	1		1		1		1		1		1		1								1				<=	1
		22		1		1		1		1		1		1		1								1			<=	1
		23	1		1		1		1		1		1		1										1		<=	1
		24		1		1		1		1		1		1		1										1	<=	1
		25	1	1	1	1	1	1	1	1	1	1	1	1	1	1											=	1
		26															1	1	1	1	1	1	1	1	1	1	=	1
	(1)		1		1		1		1		1		1		1		7		7		7		7		7		<=	7
	(2)		1		1		1		1		1		1		1		1		1		1		1		1		<=	1

Figure 5: Summary of adding and lifting constraints

Through experimentation it is realised that pair wise model is reduce to x constraints. x can be computed by taking the greatest common divisor gcd of two measurands. In our example the gcd of 14 and 10 is 2, therefore the pair wise model reduce to 2 constraints.

This technique is applied to all the pair wise view and the model is reduced to the one shown in figure 6. This formulation is name as model II.

1		1		1		1		1		1		1		1		1		1		1																								<=	1
	1		1		1		1		1		1		1		1		1		1		1		1																					<=	1
1		1		1		1		1		1		1												1		1		1		1		1		1		1		1		1		1		<=	1
	1		1		1		1		1		1		1												1		1		1		1		1		1		1		1		1		1	<=	1
														1										1										1										<=	1
															1										1										1									<=	1
																1										1										1								<=	1
																	1										1										1							<=	1
																		1										1										1						<=	1
																			1										1										1					<=	1
																				1										1										1				<=	1
																					1										1										1			<=	1
																						1										1										1		<=	1
																							1										1										1	<=	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1																															=	1
														1	1	1	1	1	1	1	1	1	1																					=	1
																								1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	=	1

Figure 6: Reduced Model

4 Results

Experimental data suggest the 5% of the time when model I gives feasible LP solution, model II gives infeasibility. Emphasising strengthening. However, in some case when model I give infeasibility it is possible to get feasible LP solution for model II. This is because of the pair wise nature of the model. At this stage results are still getting generated and detail summaries of the experimentation will be summarized and presented in the conference.

5 References

Herley-Metraplex, 1998, *PCM Telemetry Standards, http://www.metraplex.com /irigpcm.htm*

Horan Horan, S, 1993, *Introduction to PCM Telemetry Systems*, CRC Press Inc., Boca Raton, Florida USA.

IRIG Telemetry Group, May 1996, *Telemetry Standards*, Range Commanders Council, U.S. Army White Sands Missile Range, NM, IRIG Standard 106-96.

P. David, C. Stephen, 2001, *Flight Test Data Cycle Map Optimisation*, University of South Australia, South Australia.

S. Martello and P. Toth, 1990, *Knapsack Problems: Algorithms and Computer Implementations*, Chichester, NY: J. Wiley and Sons.