A Weighing Algorithm for Multihead Weighers

James N. Keraita [1] and Kyo-Hyoung Kim[2,#]

1 Graduate School of Mechanical Engineering, Yeungnam University, Gyeongsan, South Korea
2 School of Mechanical Engineering, Yeungnam University, Gyeongsan, South Korea
Corresponding Author / E-mail: khykim@yumail.ac.kr; Tel: +82-53-810-2444; Fax: +82 53 813 3703

KEYWORDS : Multihead weigher, Combinatorics, Bit operation, Accurate weighments

In industry, multihead automatic combination weighers are used to provide accurate weights at high speed. To minimize giveaway, greater accuracy is desired, especially for valuable products. This paper describes a combination algorithm based on bit operation. The combination method is simple and saves time, since only the elements to be considered for combination are generated. The total number of combinations from which the desired output weight is chosen can be increased by extending the combination from memory hoppers to include some weighing hoppers. For an eight-channel weigher, three or four combination elements are best. In addition to targeting approximately equal amounts of products in each channel, this study investigated other schemes. Simulation results show that schemes targeting combination elements with an unequal distribution of the output weight are more accurate. The most accurate scheme involves supplying products to all memory and weighing hoppers before commencing the combination operation. However, this scheme takes more time.

Manuscript received: December 12, 2005 / Accepted: April 18, 2006

NOMENCLATURE

k = number of combination elements
n = number of channels
N = total number of combinations

1. Introduction

Since the introduction of multihead weighers in the 1980s, the principle of selecting and combining the optimum number of part-weighments from a number of weighing hoppers to give a precise weight has become widely accepted, especially in the food industry. The constant dilemma for both the manufacturers and users of automatic weighing machines is the trade-off between speed and accuracy. The trade-off point is influenced by legal requirements. Average weight legislation, such as 500±3 g, implies that all of the weights (or at least a specified percentage) must lie between 497 and 503 g. Once the legal requirements have been satisfied, a greater accuracy is desirable to minimize product giveaway, although high accuracy weighing typically comes at the expense of reduced speed, which implies reduced output and efficiency.

When multihead weighers were first introduced into the market, they were so much faster and more accurate than their linear predecessors that the packaging machine being fed by the weigher suddenly became the factor limiting output, instead of the weigher. During the intervening years, however, packaging machine speeds have caught up and now the focus is again on weighing.[1]

Research has studied the chaotic dynamics (sliding and hopping) of vibratory conveying in order to increase the transportation rate of the products before weighing.[2–4] In general, the transport rate is difficult to calculate in a purely analytical manner. However, the hopping regime provides the highest rate of conveying. Han and Lee[4] identified this regime in a numerical simulation and experimental analysis. Conversely, accuracy has been enhanced by increasing the number of channels and improving signal-processing technologies. processing (DSP) and anti-floor vibration (AFV) features have been developed.[5] Fiber optic load cells are also being considered as a replacement for wire and metal foil gauges[6] due to their suitability for use at elevated temperature and immunity to electromagnetic and radio frequency interference.

In addition to speed and accuracy, there are other factors to consider in optimizing the performance of automatic weighing systems. For example, although initially developed for dry goods, multihead weighers are now increasingly being used for wet products.[7–9] Therefore, the inclination of the vibrating feeders and hoppers needs to be considered. Maintenance, cleaning, and product changeovers should be carried out easily in minimum time. All pre-feed and hoppers should preferably be removed and replaced without using tools.

Further developments need to be made for combination weighers. The ability of an automatic weigher to control product flow, cope with product variation and speed, and separate the product into accurate weighments is crucial for overall performance. This paper proposes a method for increasing the number of combinations (without modifying the hardware) of weighments from which the combination that best totals the desired output weight is selected. It develops a combination algorithm based on bit operation. The combination method is desirable because only the elements to be considered for combination are generated. Finally, the effect of the channel setup values on accuracy is investigated.

2. Hardware Layout and Operation

Fig. 1 shows the arrangement of the feeders and hoppers in an automatic combination weigher. In this setup, eight weighing channels are arranged uniformly around a circle. Each channel is equipped with a line feeder, preliminary hopper (PH), weighing hopper (WH), and memory hopper (MH). The 24 hoppers have doors operated separately by air cylinders. The advantages of using air cylinder activation include high-speed hopper opening, robust resistance to high-pressure cleaning, low heat transfer, and inexpensive and easy maintenance.

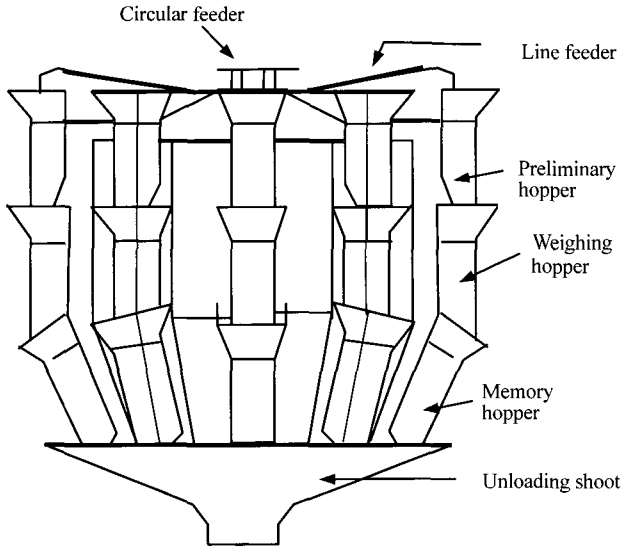

Fig. 1 Arrangement of feeders and hoppers

The products are fed through the circular feeder located at the center of the upper plate of the combination weigher. The circular feeder, which is operated by a magnetic field, distributes products to each of the eight preliminary hoppers through the line feeder. The line feeder approximates the operating time by analyzing the relationship between previous operating times and the amount of product delivered to the hoppers using the least squares method. This method is superior to using level sensors[7,8] or setting a fixed time[9,10] because most products encountered have irregular shapes or specific weights and the exact transport rate is difficult to determine.

When a certain amount of product (approximately equal to the channel setup values) is supplied to the preliminary hoppers, both feeders stop, and the contents of the preliminary hoppers are transferred to the weighing hoppers. The products are then weighed and transferred to the memory hopper immediately. The weight of the products in the memory hoppers is used in the combination process to attain the target value. Products from the selected hoppers are then released to the packaging machine via discharge chutes. New products are promptly fed into each of the emptied hoppers, thereby continuing the weighing operation.

Fig. 2 illustrates the basic principle of combination weighing for eight channels with four combination elements.

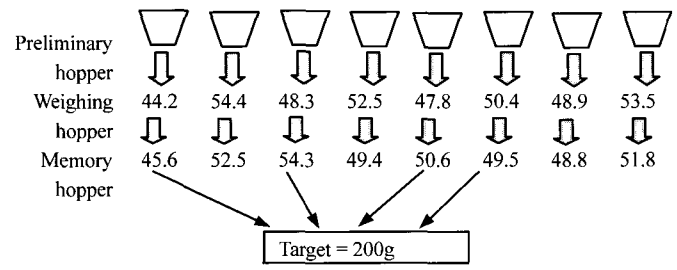

Fig. 2 Basic combination principle

3. Combination Algorithm

3.1 Total Number of Combinations

A combination is an unordered selection made from a group of objects. The possible combinations of n things taken k at a time is given by

$$ {}^{n}C_{k} = \frac{n!}{k!(n-k)!} \tag{1} $$

For multihead weighers, n is the number of channels and k is the number of weighments (combination elements). Therefore, in an ordinary combination, k is the number of memory hoppers whose contents are being combined to provide the desired output weight. However, the total number of combinations considered here is slightly different. To increase the number of combinations beyond ${}^{n}C_{k}$, the weighing hoppers are promptly resupplied with products after the previous contents are released into the memory hoppers, before the combination operation commences. The combination can then be extended to include the weighing hoppers. The weighing hoppers to be considered must be in line with the chosen memory hoppers because a weighing hopper cannot be emptied before the corresponding memory hopper is emptied.

To save time, the combination is first obtained by combining only memory hoppers, and the number is ${}^{n}C_{k}$. If the desired output is not achieved, then the combination is extended to the weighing hoppers. Fig. 3 shows the operational flowchart.

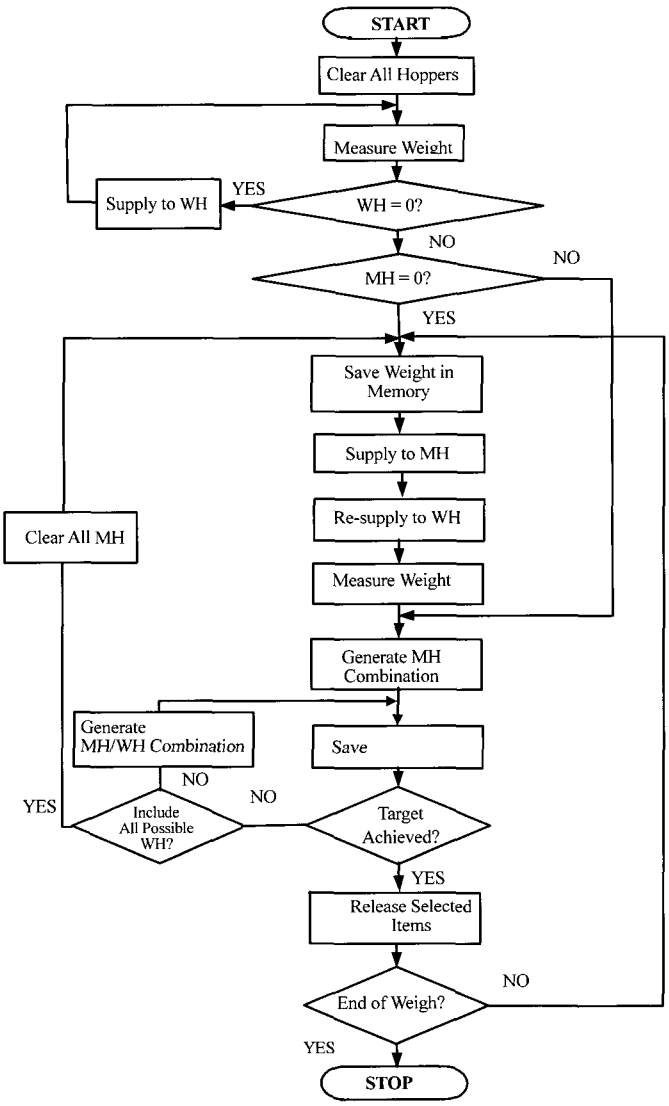

Fig. 3 Operation Flowchart

If the combination includes a single weighing hopper, the number of extra combinations achieved is ${}^{n}C_{k-1}({}^{k-1}C_1)$. If exactly two weighing hoppers are considered, then the number of extra combinations can be expressed as ${}^{n}C_{k-2}({}^{k-2}C_2)$. Since the number of weighing hoppers included in the combination can be varied, the total number of possible combinations is the summation, *i.e.*,

$$N = {}^{n}C_{k} + {}^{n}C_{k-1}\left({}^{k-1}C_1\right) + {}^{n}C_{k-2}\left({}^{k-2}C_2\right) + \cdots$$

$$\therefore N = \sum_{i=0}^{k/2} {}^{n}C_{k-i}\left({}^{k-i}C_i\right) \tag{2}$$

Table 1 compares the ordinary combination with the extended combination that includes weighing hoppers for an 8-channel weigher.

Table 1 Comparison between an ordinary combination and an extended combination

Number of combination elements, k	N (Ordinary combination)	N (Extended combination)
2	28	36
3	42	112
4	70	266
5	42	448

For an ordinary combination, three or four elements are obviously desirable because they result in a high number of combinations to choose from, hence leading to greater accuracy. Choosing five elements results in 42 combinations, as does choosing three elements. However, the code to generate 5-element combinations is less efficient.

For the combination including weighing hoppers, a greater number of combination elements might result in a greater number of total combinations. However, the next cycle following the one in which the weighing hoppers have been emptied will result in fewer logical combinations. This is because the weighing hoppers will not transfer any product to the corresponding memory hoppers. If a large number of weigh hoppers are emptied, the logical combinations can be reduced drastically. Consequently, this effect, together with time wasted on generating illogical combinations (i.e., involving zero weight in memory hoppers), three or four combination elements are the most desirable, even when extending the combination to include weighing hoppers.

3.2 Method of Generating Combinations

A mathematical combination lends itself very nicely to implementation as a class in C++. For data members, one needs to store the values of n (total number of items) and k (number of items in each subset element), and an array to hold the individual integers of each combination. The combinations and individual integers of a combination element should appear in a lexicographic order to avoid repetitions such as [1234], [1342], or [4321].[11] There should be a kind of dual orderliness. All combinations can be made, albeit rather inefficiently, using a series of nested 'for' loops.

These techniques work well if one wants to generate all the elements of a combination when n and k are small numbers. As n and k increase, generating the combinations takes appreciable time. Additional time is required if one wants to find the best weighted subset of k things out of n, as in the case of combination weighers when a target output weight is sought.

Therefore, this paper suggests a combination method using bit operation. The principle of the bit operation is to correspond each element of the combination to one bit. If the bit has a value of 1, then the element is in the set, otherwise the element is not in the set. For example, the following binary number: 11010100 with bit numbers running from 7 down to 0, represents the set: {7 6 4 2}. Therefore, ${}^{n}C_{k}$ translates into generating all the binary numbers of length n with exactly k 1-bits. A method commonly used to do this is to generate all possible n-bit numbers, count the set bits (1s) in each, and print the corresponding combination when the number of set bits is equal to k. This method is simple to understand and implement, but is very inefficient. A more efficient way is to be able to create a combination with k set bits and then determine its lexicographic successor, which also contains k set bits.

To understand the procedure, let us consider ${}^{8}C_{4}$. The smallest 8-digit binary number that contains exactly four set bits is 00001111, while the largest is 11110000. To generate the first combination, we use $(1UL << k) - 1UL$, where UL is a variable of unsigned long type and $<< k$ means a bitwise shift to the left by k positions. A look at the two extreme binary numbers reveals a mirror image relationship. A bitwise shift by $(n - k)$ positions to the right makes the first number equal the last. This gives us an idea for the stop condition.

The algorithm for generating the combinations is as follows:

1. Current x
2. Find $-x$
3. $s = x \,\&'\, -x$

   The operator &' is slightly different from a bitwise AND (&). The bit in the result of x & $-x$ corresponding to the first high bit in x from the extreme right is always set to 1.
4. $l = x + s$
5. $p = l \,\&'\, -l$
6. $q = \left(\left(\frac{p}{s}\right) >> 1\right) - 1$

   Division by $s = 2^{m}$ is equivalent to a bitwise shift to the right by m positions.
7. Update x. $x = l|q$; *i.e.*, a bitwise OR.

Fig. 4 shows the flowchart for implementing the combination with bit operation.

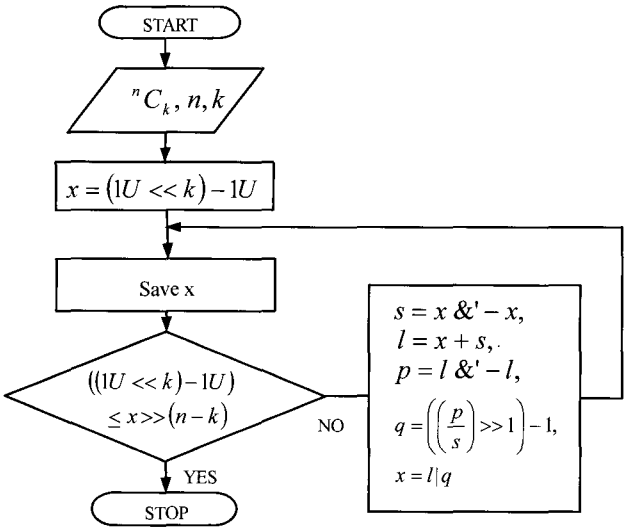

Fig. 4 Flowchart for combination with bit operation

The algorithm is easily implemented when combination is extended to include weighing hoppers. If the already generated memory hopper (MH) combinations are considered individually for possible combinations with weighing hoppers, the problem is reduced to that of deselecting r (r = 1, 2 ..., k/2) memory hoppers and replacing them with an equal number of weighing hoppers that are in line with the still selected memory hoppers.

4. Results and Analysis

If a number of memory hopper contents are to be combined to give a desired output, one could, by intuition, target an approximately equal distribution among the hoppers during each cycle. However, this might not be the best way to continue achieving high accuracy of the desired output weight over time. Table 2 shows five different schemes

for the computer simulation investigated in this research. The amount of product targeted in each line is the channel setup value. As discussed in section 2, it is difficult to supply weights equal to the channel setup values accurately due to the irregularity of the products and the chaotic transport mode, which involves sliding and hopping.

Table 2 Schemes for the computer simulation

Scheme	Channel Setup value (g)								No. of Elements, k	Output Weight (g)	Note
	1	2	3	4	5	6	7	8			
A	50	50	50	50	50	50	50	50	3	150	
B	50	50	50	50	50	50	50	50	4	200	
C	40	40	50	50	50	50	60	60	3	150	
D	40	40	50	50	50	50	60	60	4	200	
E	40	40	50	50	50	50	60	60	4	200	Exact Mode

Schemes A and B both use channel setup values of 50 g in all channels, while using 3 and 4 combination elements, respectively. Hence, the desired output weight is 150 g for scheme A and 200 g for scheme B. Schemes C and D are variations of schemes A and B that utilize unequal channel setup values. Scheme E (Exact Mode) is similar to scheme D in terms of the channel setup values, but in this case, the empty weighing hoppers caused by the extended combination in the previous cycle were supplied with products to avoid zero weights being transferred to the corresponding memory hoppers. After weighing and transferring the products to the memory hoppers, the weighing hoppers were resupplied with products before the combination operation commenced.

The simulation results for each scheme (A–E) are as shown in Figs. 5–9.

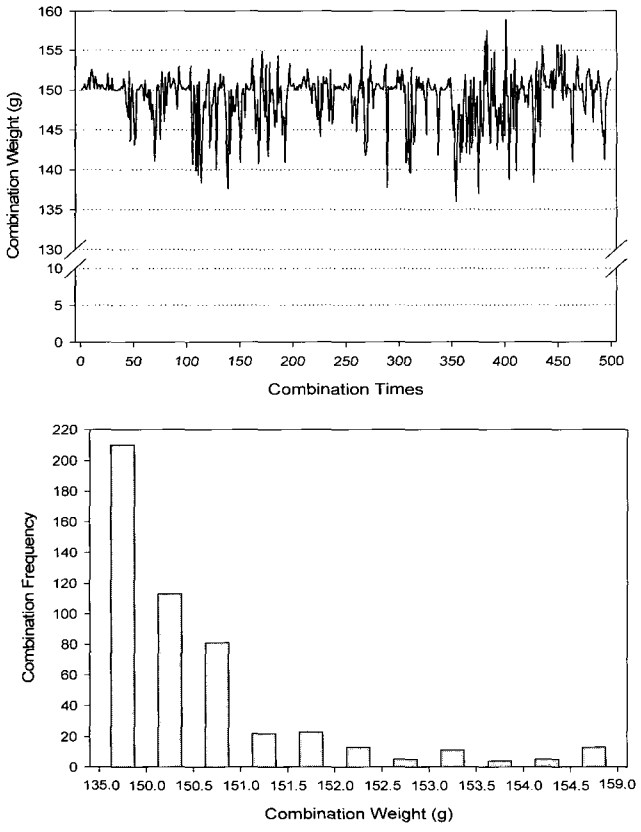

Fig. 5 Simulation results and histogram of scheme A

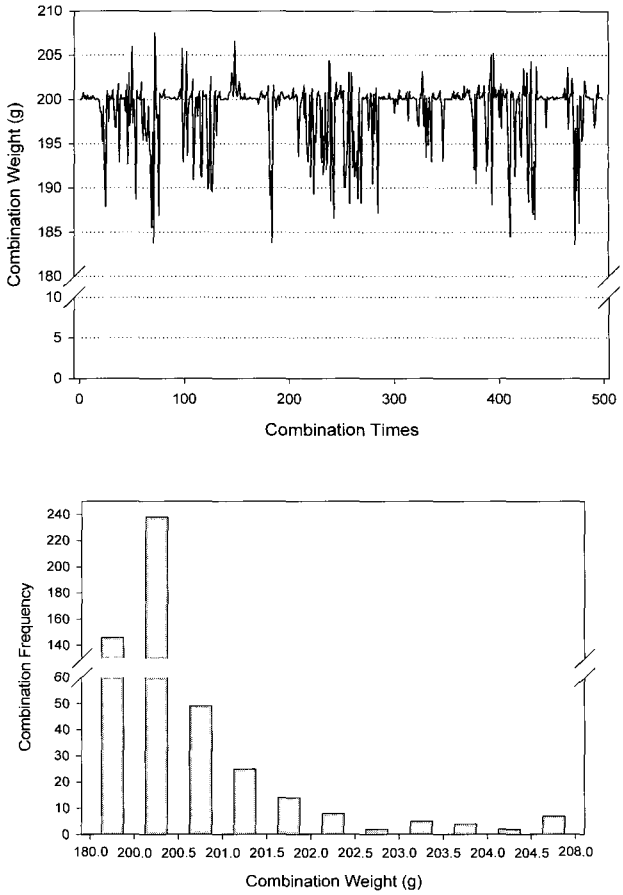

Fig. 6 Simulation results and histogram of scheme B

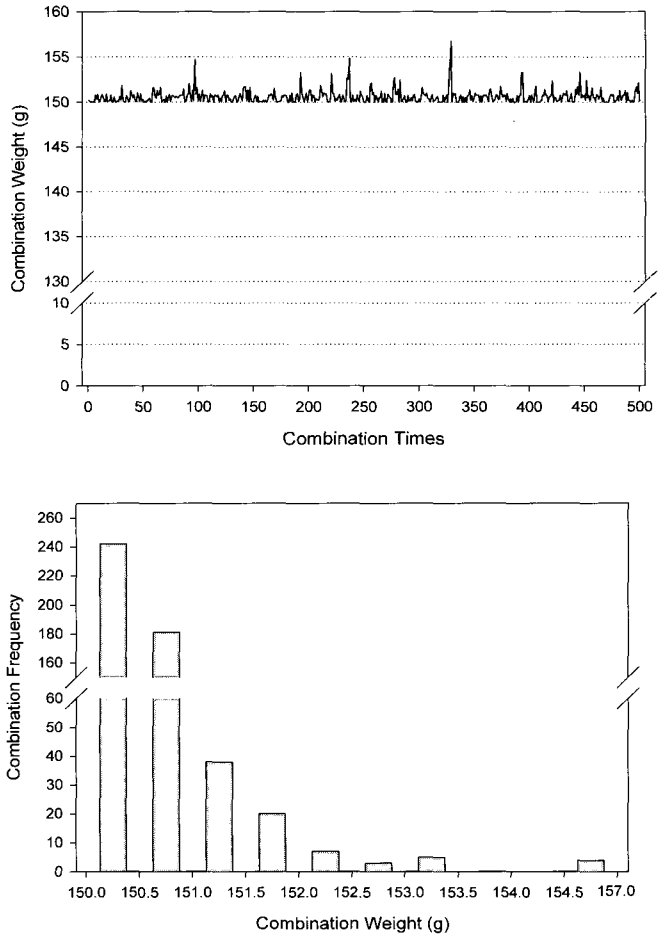

Fig. 7 Simulation results and histogram of scheme C

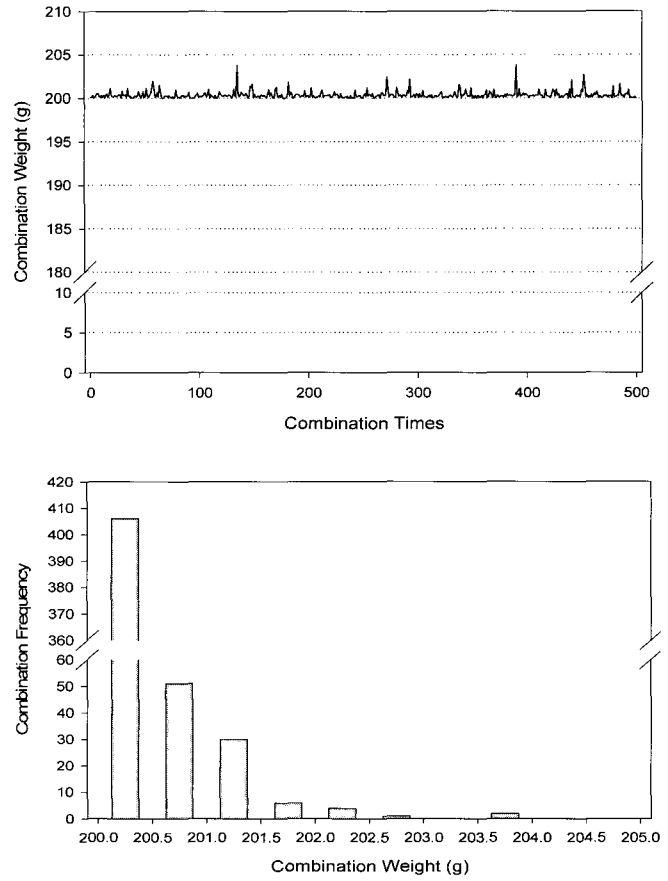

Fig. 8 Simulation results and histogram of scheme D

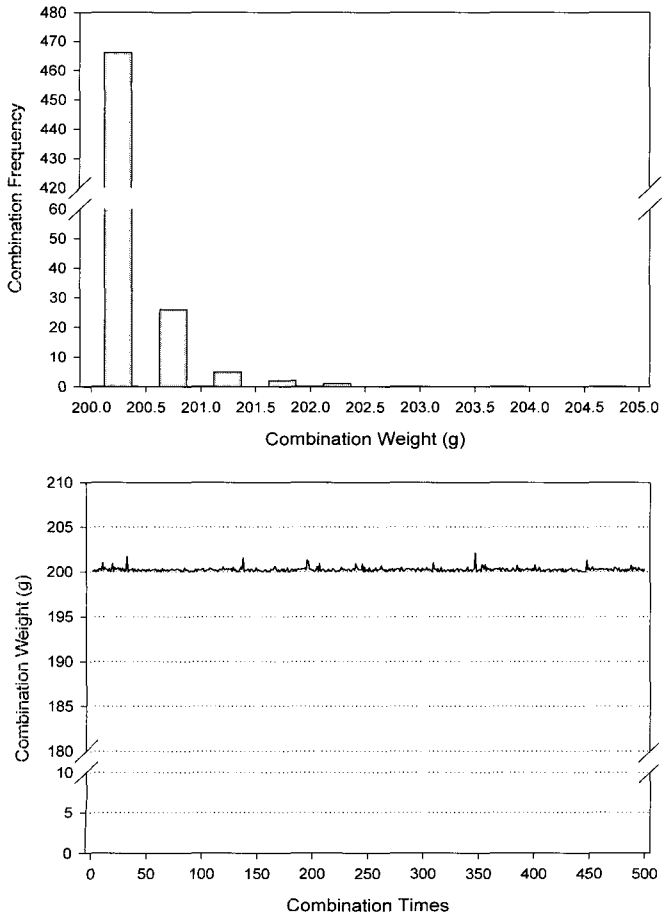

Fig. 9 Simulation result and histogram of scheme E

Unacceptable high-precision errors were found for the case of equal channel setup values (schemes A and B), as shown in Figs. 5 and 6. The desired weight is only closely achieved in the first period, after which the error increases significantly. These schemes assume that the products supplied to some channels will always be overweighted, while the rest will be underweighted, whenever exact channel values are not achieved. However, the products supplied to all channels can easily be overweighted or underweighted since in the short period of feeder operation, the properties of the batch supplied are essentially the same.

To avoid errors caused by this phenomenon, schemes C and D were proposed. In these schemes, the first two channels were set at 40 g, while the last two were set at 60 g. The remaining intermediate channels were maintained at 50 g. The idea was to ensure that a suitable combination could still be obtained even in cases when all the products supplied to the hoppers are either overweighted or underweighted. The results (Figs. 7 and 8) show an appreciable improvement over schemes using equal channel setup values.

Scheme E (Exact Mode) ensures that no illogical combinations in which some memory hoppers have zero weights are encountered. This scheme is the most accurate (Fig. 9). However, because products have to be supplied to some channels twice before the combination operation, it takes considerably more time. The extra time needed depends on the number of channels in which both the weighing hopper and memory hopper contents were included in the best combination of the previous weighing cycle.

Table 3 shows an additional statistical comparison of the results. Accurate weights were deemed to have been achieved if the discrepancy from the output target was less than 1 g. This is an error of 0.5% for the 200 g output targets and 0.67% for the 150 g targets. For 500 tests, the Exact Mode produced 98.4% accurate results, while the equal channel setup value scheme produced as low as 38.8%. Proper error analysis is important in estimating the performance of a precision machine.[12]

Table 3 Statistical comparison of the results of the five schemes

Item	Scheme A	Scheme B	Scheme C	Scheme D	Scheme E
Output Target (g)	150	200	150	200	200
Max (g)	158.95	207.55	156.75	203.80	202.12
Min (g)	136.00	183.55	150.03	200.05	200.00
Accurate Weights	194	287	423	457	492
Probability (%)	38.80	57.40	84.60	91.40	98.40

5. Conclusions

The accuracy of multihead weighers can be enhanced by increasing the total number of combinations from which the best combination (one that most accurately results in the desired output weight) can be chosen. In this study, the number was increased appreciably by extending the combination from memory hoppers to include some weighing hoppers. Our combination algorithm based on bit operation is simple and saves time, since only the elements to be considered in the combination are generated.

Although it may appear logical to target an equal distribution of the desired output weight among the hoppers, the results show that schemes targeting an unequal distribution of weight are more accurate. The mode in which all the memory and weighing hoppers are supplied with products before the combination operation commences is most accurate. However, this mode (Exact Mode) takes more time. Exact Mode is recommended for high-value products, to minimize

giveaway, especially with the emerging trend in which the customer is always given the benefit of overweight.

REFERENCES

1. Keay, M., "Improving the multihead," Machinery Update, pp. 38-42, 2005.

2. Hongler, M. O. and Figour, J., "Periodic versus chaotic dynamics in vibratory feeders," Helvetica Physica Acta, Vol. 62, pp. 68-81, 1989.

3. Lim, G. H., "On the conveying velocity of a vibratory feeder," Computers and Structures, Vol. 62, pp. 197-203, 1997.

4. Han, I. and Lee, Y., "Chaotic dynamics of repeated impacts in vibratory bowl feeders," Journal of Sound and Vibration, Vol. 249, No. 3, pp. 529-541, 2002.

5. Maddox, M., "Higher speed in prospect," Machinery Update, pp. 60-63, 2001.

6. Mastro, S. A., "The effect of transverse load on Fiber Bragg Grating measurements," M.S. Thesis in Materials Engineering, Drexel University, 2000.

7. Ishida Europe: http://www.ishidaeurope.com/

8. Anritsu Corp.: http://www.anritsu-industry.com/E/

9. Bilwinco Weighers: http://www.bilwinco.com/

10. US Patent: http://www.freepatentsonline.com/4967383.html

11. McCaffrey, J., "Using combinations to improve your software: test case generation," MSDN Magazine, July 2004.

12. Lim, S. R., Choi, W. C., Song, J. B. and Hong, D., "Error model and accuracy analysis of a cubic parallel device," Journal of the KSPE, Vol. 2, No. 4, pp. 75-80, 2001.