CHAPTER 5: ANALYSIS OF COORDINATION TECHNIQUES ON COMBINATIONS OF APPLICATION SYSTEMS

The coordination and scheduling techniques are put into practice and the upshots are analyzed by operating it in five different applications namely linear heating, clustering, cloud service, pattern searching and infant incubator. These five applications are combined to form 2 combinations. Each combination contains 3 applications. The combinations are selected to analyze the characteristic outcome of the application, when scheduled with similar or dissimilar applications.

Scheduling is the management of CCHBs to meet the performance improvement according to not only the gain factor but also the execution order of CCHBs, because dependency is based on the execution order. All the identified profitable CCHBs of different applications are scheduled and configured in the FPGA eventually and hence simple mapping is used to assert whether scheduling has given precedence to the profitable independent CCHBs or not. During the scheduling process, the device utilization reduction is one of the key performance factors and FPGA size is used to decide which CCHBs are accommodated in current configuration. To study the performance of the proposed mechanism, the mathematical model given below is used.

5.1 MATHEMATICAL MODEL

The placement of CCHBs of five different applications into FPGA is modeled mathematically as a knapsack problem. The Gain Factors (GF) based on the attributes of CCHBs such as CCHB-FPGA size ratio, speedup, software cycles saved, occurrence rate and merging factor are designated as weights of the CCHBs. With the maximum weight,
the CCHBs of five different application systems are accommodated into the available FPGA size as knapsack capacity.

\[ \text{Max } f(\text{CCHB}) = \sum_{i=1}^{n} \text{GF of } \text{CCHB}_i \]

subjected to

\[ \text{FPGA size} \geq \sum_{i=1}^{n} \text{size of } \text{CCHB}_i \]

The set of feasible solutions is given by

\[ X = \left\{ x \in \text{INT}^n : \text{FPGA size} \geq \sum_{i=1}^{n} \text{size of } \text{CCHB}_i \right\} \]

The feasible solutions are generated using optimization tool with branch and bound technique. The technique involves two functions called branching and bounding. Branching is a function that divides the problem into smaller sets whose combined result forms the solution to the complete problem. Bounding is a function that defines the lower and upper bound for the minimization or maximization. The main feature of branch and bound technique is their tree structure and the process of pruning that discards the tree node based on the bound values.

The five application systems are framed as knapsack problem individually and each problem is solved using the lingo optimization tool [108]. The solution status of linear heating and infant incubator applications is illustrated in figure 5.1. Similarly solution status of clustering and text pattern searching applications is demonstrated in figure 5.2 and of cloud service application in figure 5.3. It is observed from the figures 5.1 to 5.3 that the model class of solver is linear programming, state of the solver is global optimum, solver type is branch and bound, number of constraints is 2 (maximum gain and occupying less logic blocks), number of variables (CCHBs) is different for each
application. The list of selected CCHBs from the optimization tool is verified with that of proposed coordination techniques for their correctness. It is observed that the average accuracy percentage of the proposed scheduling for five applications individually is around 99.9% with that of the optimization tool.

Figure 5.1 Solution status of linear heating and infant incubator applications

Figure 5.2 Solution status of clustering and text pattern searching applications
5.2 COMBINATIONS OF APPLICATION SYSTEMS

Applications are combined to study the resultant metrics of proposed mechanism, when an application is scheduled with similar or dissimilar applications. The applications are scheduled by fixing FPGA size in the scheduler. The resultant metrics are computed and analyzed with the following application combinations.

The applications are combined as

i. Combination 1:
   a. Incubator and Cloud service,
   b. Cloud service and Pattern searching,
   c. Pattern searching and Incubator,
   d. Incubator, Cloud service and Pattern searching.

ii. Combination 2:
   a. Linear heating and Clustering,
   b. Clustering and Incubator,
c. Incubator and Linear heating,

d. Linear heating, Clustering, and Incubator.

AS1 and AS2 are any two applications among five application systems. In this work, in combination 1a, processing time of incubator (AS1) when scheduled with cloud service (AS2).

5.2.1 COMPUTATION OF RESULTANT METRICS

Resultant metrics are computed in % as follows:

Performance ratio of proposed mechanism against random scheduling =

\[
\frac{\text{Processing time of the AS1 when scheduled using proposed technique with AS2}}{\text{processing time of AS1 when scheduled using random scheduling technique with AS2}} \times 100.
\]

-Eqn 1

Performance ratio of proposed mechanism against First Come First Served (FCFS) scheduling =

\[
\frac{\text{Processing time of the AS1 when scheduled using proposed technique with AS2}}{\text{processing time of AS1 when scheduled using FCFS scheduling technique with AS2}} \times 100.
\]

-Eqn 2

Software execution time saved by proposed mechanism =

\[
\frac{(\text{total execution time of AS in software} - \text{software execution time excluding hardware part})}{\text{total execution time of AS in software}} \times 100.
\]

-Eqn 3

Cumulative utilization of proposed mechanism =

\[
\frac{\text{Total logic cells occupied by AS in various rounds}}{\text{total FPGA logic cells available for configuration}} \times 100.
\]

-Eqn 4
Increase in speed of proposed mechanism = 

\[
\frac{(SW \text{ exe. time of HW part} - HW \text{ exe. time of HW part})}{SW \text{ exe. time of HW part}} \times 100
\]

-Eqn 5

Where, FCFS is first come first served, SW is software, HW is hardware and processing time is the execution time of application.

5.3 RESULTS AND DISCUSSION

The scheduling and coordination technique is developed in C language and carried out experiments on five real time applications to study the performance of proposed scheduling. The computation intensive parts of the applications are designated as CCHBs from the DFG of the application using Trimaran compiler, profitable independent CCHBs are identified using conflict graph approach, and occupancy of FPGA is further reduced by data path merging using compatibility graph approach.

The resultant metrics of proposed scheduling for combination1 and combination2 are computed using Eqn 1 through Eqn 5 and are illustrated in figure 5.4 and 5.5 respectively. It is observed from the figure that the average performance improvement is approximately 60-80% against other scheduling mechanisms. The average software execution time saved is nearly 67% and average increase in speed is around 82% against full software implementation. The device utilization is about 24% of FPGA which is satisfactorily less when compared to complete FPGA implementation. The selected CCHBs of the combinations are determined through lingo optimization tool and it is found to agree with that of proposed scheduling.
Figure 5.4 Result analysis of proposed scheduling for combination 1

Figure 5.5 Result analysis of proposed scheduling for combination 2
The proposed mechanism is compared with other scheduling mechanisms such as random order and FCFS. The proposed framework is always looking for the best ones and identifies profitable CCHBs. The baseline algorithms will identify the profitable CCHBs only when it comes into the way when selecting with random order and only when it comes in first when selecting with FCFS. Hence qualitatively the resultant metrics except performance ratio for those two baseline algorithms may not give better values.

The resultant metrics obtained for random order scheduling are depicted in figure 5.6 and 5.7. From the figures, the software execution time saved and increase in speed are found to be lesser for random order scheduling than the proposed mechanism. The average device utilization is about 33% which illustrates occupying more FPGA space than proposed mechanism. Figure 5.8 and 5.9 demonstrates the resultant metrics obtained for FCFS scheduling. It is observed that the software execution time saved and increase in speed are sufficiently decreased when compared to that of proposed mechanism. The device utilization is nearly 34% which shows more occupancy of FPGA than that of proposed mechanism.
Figure 5.6 Result analysis of random order scheduling for combination 1

Figure 5.7 Result analysis of random order scheduling for combination 2
Figure 5.8 Result analysis of FCFS scheduling for combination 1

Figure 5.9 Result analysis of FCFS scheduling for combination 2
The final analysis is to study the performance of an application when scheduled with other similar and dissimilar applications. The infant incubator application is scheduled with all other four applications for this study. The observed and computed results of FPGA accelerated infant incubator application are demonstrated in figure 5.10. It is inferred from the figure that there is an improvement in performance, speed and device utilization when scheduled with similar application.