Data Replication optimization using Simulated Annealing

Chee Keong Wee¹, Richi Nayak²

¹School of Electrical Engineering & Computer Science, Science & Engineering Faculty, Queensland University of Technology, Brisbane, Queensland, Australia
ckwee@outlook.com, r.nayak@qut.edu.au

Abstract. Data replication is ubiquitous in a large organization where multiple IT systems need to share information for their operation. This function is usually fulfilled by an enterprise replicating software that is dependent on the configuration that the IT administrator sets. The setup specifies the tables and routes; however, it may not be optimum to meet the workload, leading to replication’s lag and bottlenecks. This paper proposes an approach to solving the configuration optimization problem for the data replication software with the simulated annealing (SA)-based heuristic. Empirical results show that the configuration setting enables the replicating software to perform at least 5 times better than the baseline configuration set achieved by this approach.

1. Introduction

Data replication is an essential business need where data is copied to other systems to maintain high availability, data reporting, business consolidation, workload sharing as well as support disaster recovery standby nodes with redundancy in data sources [1]. In order to achieve high volume and speed transfer of data, productivity tools such as data integration and replication software are commonly used to serve the need of copying data from sources and transform them before applying them into destination systems [2]. IT administrators must manage the data replicating environment in response to the constantly growing volume of data on enterprise IT systems. With the exponential increase in the landscape of databases and their sizes, it is a challenge to ensure that these data changes can be replicated to the receiving systems quickly.

Prior research has shown that delays in the data replication occur frequently due to software’s performance constraints and thus creating bottlenecks that inhibit the software overall efficiency[3, 4]. Some of the factors that affect the process are [5]: 1) tables’ attributes do limit the number of parallelized query processes that can be made; 2) a large number of tables that needs to be involved in the replication; 3) the volume of the data that changed among the tables; 4) the different type of data that these participating tables’ columns have; and 5) the velocity, variety, and volume of SQL statements applied to tables involved in replications. A common approach is to use an optimizing technique to control the parameters and configuration of the data replication’s setup[3, 4]. For example, to effectively share the workload, the transfer rate can be manipulated by controlling the configuration of data routing queues and tables [3].

A queue represents the channel in which the data is extracted and passed from the source to the destination. If the requirement is of transferring the high volume of data, all the changes that passed to the queue for transfer will build up. This may create a bottleneck if the transfer queue does not process the replication fast enough to clear the
backlog. It is not desirable to have a series of tables that have a different level of low and high post activities congregate around specific queues. While some queues can clear tables with low activities and then go into idle mode, others will have a lot of highly active tables and create huge backlogs, hindering the overall replication [1, 3]. From the IT administrator’s perspective, an imbalance queue will delay the data transfer that creates an incomplete point-in-time view of the source database at the destination. This will delay the necessary operation that is dependent on it [2].

A possible solution is to create more queues to support the replication; however, there are several shortcomings with this approach[3]. Each queue requires additional system resources to run. The more queues that the replicating setup has, the more resource is required to support them as well an overhead will be incurred. It is desirable that replicating tables with a range of low to high changing activities should be arranged across a predefined number of queues so that the transfer of data changes for a given volume can be cleared at the least time.

We propose a novel method to optimize the configuration for the data replicating setup based on a Simulated Annealing (SA) algorithm[6]. There is no prior research made in optimizing the performance of data replications in DB environment to our best knowledge. The computational search space to find the best arrangement of queues with tables of varying workloads is high and it is difficult to work through all the different combinations. Techniques like gradient descent and statistical methods will take a long time to solve this problem. we want to find the near-best solution within a short time limit. SA is metaheuristic and it can approximate global optimization for the data replication problem since there is a large search space. The outcome is a configuration of the best arrangement of tables with different levels of activities against a pre-defined number of queues.

Each iteration of the test was made with the activation of a unique Shareplex’s configuration arrangement, followed by running a procedure that performs a variety of SQL changes on the tables at the source site. The overall duration for the entire SQL batch update to be transmitted across from the source to the target site using that configuration set was measured. Each unique set of configurations enables Shareplex to work differently, and with SA algorithm to test and search through a list of different configurations. Empirical results show that a near-global optimum configuration was found to satisfy the DB replication’s data transfer need. To our best knowledge, this is the first method based on the AI technique to optimize the performance of the data replicating tool like Shareplex.

2. Related works and background

Data replication serves different needs in IT organizations such as data distribution, workload sharing, reporting, backup or disaster recovery [1, 7]. For data distribution, the key requirement is that data must be shared quickly across a multitude of systems with minimum delays. For workload sharing, the key requirement is to increase a system’s throughput by distributing its operation across a multitude of hardware to ease the workload. Another use for fast data replication is to keep an auxiliary copy of the primary data as a form of backup in case the primary data source suffers a catastrophic physical failure or logical corruption in order to maintain a high level of uptime and
availability. In all cases, the data must be reconciled at near real-time to protect the data integrity and quality, so the changes in the data must be reflected at the repository which resides in a different location.

A survey paper [8] presented a series of time and space-based strategies used in data replication deployment, taking into consideration the characteristics and requirements of the data and systems. The survey covered the consideration of selecting the appropriate files for replication, emphasizing important and most accessed groups over those with low demands. They also proposed performance evaluation metrics to evaluate each of the different strategy's effectiveness[8]. Recently, a new evaluation metric was proposed for measuring the execution time and bandwidth consumption of data replication leading towards optimization [9]. Authors in [10] proposed the centralized dynamic scheduling and replica placement strategies that manage the data and task scheduling for optimum cost and data transfer time, in order to improve the file access time by applications across a geographically-wide data grid.

For solving optimization problems such as this, meta-heuristics is one of the more popular approach. Meta-heuristics can be classified into two groups; population-based and single solution-based [11]. The problems that they handle are either continuous or discrete, and a problem such as finding the best queue-table arrangement for Shareplex can be considered as a combinatorial optimization problem (COP) which is regarded as NP-hard where no optimum configuration for Shareplex can be found within an acceptable time and resources [11]. The possible optimum configuration is considered as a single solution that can serve the data replication environment, so the requirement is to use single solution-based algorithms such as hill climbing or simulated annealing [11]. Other meta-heuristics such as Genetic algorithm, Ant colony or particle Swam optimization are more suitable for problems that have a population of solutions[11].

On the other hand, the Simulated Annealing (SA) method has been commonly used to solve the NP-hard problems that range from transportation and energy production to system optimization. An example is a truck-trailer routing problem where each delivery route requires to serve several customers over a defined geographical region and it needs to adhere to a constraint of allocated window periods for the delivery trucks to fulfill [12]. Another example is of finding the ideal location for the turbines to be installed across the water distribution network that can capitalize on the water flow to generate hydroelectricity [13]. SA has also been applied to solve a Multi-objective Redundancy allocation problem, improving the reliability of a system which comprised of subsystems of numerous components that each have different constraints [14].

To our best knowledge, there are no articles that are devoted to the data replication optimization problem between databases with reference on specific application tools such as Shareplex[15] or GoldenGate[4]. This research work to address the optimization of data replication tool’s configuration sets precedence. Next, we describe the data replication software and the simulated annealing algorithm used in this research.

2.1 Shareplex data replication

There exist several types of data replication software specific to the type of database that it supports and the functionality that the users want. While Extract-Transform-Load (ETL) software such as SAP Data Services or IBM’s DataStage can be used for replicating data, they don’t have the capability to do it near real-time. That is where a
specialized tool such as Shareplex[3], Oracle’s GoldenGate[4] and SAP’s Smart Data Integration comes in [16]. Shareplex is used for this research as it is the most popular data replicating tool developed by Quest software for both commercial and open-source databases [17]. Shareplex runs in the background and performs data capture by reading the database’s log files for changes to tables that are listed in its replication configuration, then propagates and applies the changes over to the target systems in near real-time. The Shareplex’s framework comprises several components of data capture, read, export, import, and post, in addition to the source and target databases that they run against. Figure 1 shows how the changes are captured and transported from the source to the target databases. The major components and subsystems of Shareplex are as follows.

**Capture process** – It runs against the source database, constantly reading the redo logs and sometimes archived logs for changes. It then sends the change to the capture’s queue[18].

**Queues** – All the queues are dynamic data repositories that hold the temporary data for the duration of data capture, transmission, and reception through the process of data replication. The order of the queues relationship follows from the capture’s queue to the read’s queue, to the export, then to the target side’s import which connects to the post queue.

**Read process** – This runs only at the source site. It reads the data from the capture queues and processes it by repackaging them with information for network transmission. The processed data is then stored in the export queue.

**Export process** – This runs at the source site. It reads the processed replicated data from the export queue and transfers to the target across the network.

**Import process** – This runs at the target side. It intercepts all transported replicated data sent out by the export process and stores them in the import queue.

**Post process** – This process runs at the target side. It transforms the data read from the import queue into the relevant SQL statement before they can be executed against the target database.

![Figure 1: Shareplex’s data replication architecture](image)

**Replication configuration:** The replication can be set up or controlled by a configuration file as shown in figure 2. It defines the list of tables that need to participate in the data replication; on the source side, the schema and table_name, on the target side, the schema and object name, followed by routing information in the following format (target_system:named_queue@o.Target_oracle_sid). Shareplex can support a variety of replicating architecture; from a single, bi-direction to star topology like multiple-target and multiple-source[19].

**Shareplex filesystem:** Shareplex operates from two main directories; product directories where all the binaries are stored and executed from as well as a variable data directory where template files, licenses, parameters, logs and trace files are stored. It also stores all
the temporary data files that are used by the queues as well as the associated network routing information that is associated with them[20].

Shareplex operation: Shareplex system runs under a specific UNIX user account that shares the same admin group as the Oracle database group which was used to install the Oracle binaries. There are several UNIX’s environment shell parameters; $ORACLE_SID, $ORACLE_HOME, $SP_COP_TPORT, $SP_COP_ UPORT, $SP_SYS_VARDIR and $SP_SYS_HOST_NAME. Once Shareplex is installed on both the source and target system, the administrator will activate a configuration file to initiate the data replication. All information, including debug and errors, are captured and stored into event logs under $SP_SYS_VARDIR/logs directory. In the event should there be tables that are out of sync, Shareplex has a compare/repair feature that allows the administrator to fix the replication tables and bring them back into synchronization[20].

2.2 Simulated annealing

SA is an analogous method for optimization that attempts to find the global optima among the large landscape of local optima of solutions for a problem environment[21]. Annealing is a process where metals are heated and then cool down to a hardened condition. The objective function represents the energy of the material. SA has a similarity with a hill-climbing algorithm[21] with the exception that it doesn’t just pick the better move in its iteration but rather a random one[6]. Referring to Eq(1), if the selected move can improve the solution, it will be accepted. However, this single goal can cause the algorithm to be stuck in local optima. Hence, the algorithm also takes a chance in making a choice to accept a worse move based on some probabilities of a value that is less than 1. It starts with a high probability which means that the algorithm will be more liberal to accept the bad move but that will decrease rapidly with the degradation between the past and present moves [6].

\[
\text{If } q_{\text{limit}} < q \text{ then } p = 1, \text{ else } p = e^{\frac{\text{cost new} - \text{cost old}}{\text{temperature}}} \tag{1}
\]

The probability of accepting the uphill move is equal to, 1 - ( \( E_{\text{new}} - E_{\text{old}} / kT \)) where \( E_{\text{new}} \) is the amount of energy at the present, \( E_{\text{old}} \) belongs to the previous iteration’s energy, \( T \) determines the probability and is synonymous to the annealing's temperature. It controls the algorithm's decision to take on the hill-climbing attributes for the moves, starting from a high-temperature value of \( T \). The SA algorithm will be open to accepting the hill-climbing process. But as \( T \) decreases over time in energy, this probability will decrease, and the algorithm will be less inclined to accept this until \( T \) reaches zero. \( k \) is the constant that relates the temperature to the energy [13]. SA is commonly used to solve the optimization problem in large and discreet configuration spaces that can be considered as nondeterministic NP-complete problems such as travel salesman problem where the combination space of queues to tables arrangement is large and an approximating global solution is required with a specific period of computational time [6].
3. The Problem Definition

Contemporary data replicating software runs with processes and queues [15]. They function by constantly reading the source databases for changes. The captured data are then copied and propagate to the designated databases and are applied with minimum delays. Ideally, these changes should occur with minimum delays. However, the SQL change activities are IO intensive on the databases and will incur waits and latency on the application of the changes. This delay will increase substantially if there are huge volumes of changes that need to be replicated across, causing long delays and creating backlogs that impact on the overall replication performance [1].

In order to alleviate this challenge, prior research recommends having multiple queues running in parallel so as to spread out the entire job by breaking it into smaller sections among the queue [3]. This will reduce the load that each queue will have to serve. The approach is to mix and match tables with a different variety of Data Manipulation Language (DML) activities so that the combined tables among the queues can equal out. Usually, the highly active tables will be assigned to one set of queues and other slower ones are bundled together under other queues [5]. However, there are a finite number of queues that can be made available as more queues will equate higher system overheads and greater IT administration effort. In summary, we do not want to create thousands of queues to support the replications nor do we want to create too few queues that they are not sufficient to support the data replication effectively [2]. Another factor is that we do not know which high or low active tables can give the best combination for the replication process that can create the least amount of backlog clearing time.

The combination of tables of varying DML activities across a series of queues can be regarded as a nondeterministic polynomial time (NP)-hard problem in combinatorial optimization [5]. The objective of the research problem of data partition is to determine the tables-queue allocation plan with the timing required to clear a series of tables that will receive a different level of changes to simulate their activity in the databases. However, there exist complications that will impact the calculation. They are:

1) Tables don’t have a constant pattern. All the tables within the databases have a different level of DML activities throughout the week and some surges in data changes that coincide with peak business processes such as month-end financial processing which makes it hard to predict the volume of activities for the tables for any given period.

2) The tables vary in structures and they differ from one another in terms of column numbers, data-types and data complexities. A table with a simplified structure of multiple columns with numeric fields may have a different level of impact on the replication as compared to a table that has a few columns that store large binary objects. Some tables can span over kilo or megabytes in size per row[22].

3) While capturing the data changes at the source database can be fast, applying the changes at the target databases in SQL statements can be a challenge as it invokes a significant amount of disk IO which can be slow as compared to read actions. This slow IO latency will be exacerbated when the target tables have dependents such as indexes or queries made against them while the update is in progress. That will slow down the throughput in applying the changes significantly[1].
We must balance the load between the tables’ activities across the queues in such a way that the queues can transfer all their changes across with minimum delays and lags. We cannot treat this as a linear problem as the activities on the tables and volumes are neither linear nor consistent throughout the IT system operation [23].

4. The Proposed SA based Approach

We propose to solve the queue configuration optimization using a simulated annealing algorithm as shown in Figure 3. The data is replicated from the source to the target DB via the connecting data replicating software. For this experiment, Shareplex is used as the replicating tool. Shareplex processes the DML changes that occur on those tables that are involved in the replication at the source DB by reading the DB’s redo log files and then propagate the changes over to the target DB which has another instance of the data replicating software. The target Shareplex receives the input and applies the changes to the target DB to complete the data replication process. The mode of the replication transfer process is managed by the configuration file that specifies the queue that the Shareplex can use to channel the DML activity for the replicating tables. This configuration file has a direct impact on the performance of the overall replicating process. Referring to Table 1, one queue can service several tables, and, in some cases, it is able to support all of them if the number of tables is small and their DML activities are low. However, when the replicating system is large and their activities increase exponentially, the load will flood the queue. The queue will not have enough capacity to handle them and this will form the bottleneck in the replication process, causing massive build-up [3].

‖Source DB‖
‖Data replication‖
‖Target DB‖
‖Data replication‖
‖Optimizing agent‖

The goal is to distribute the tables across to a series of queues to spread out the load. Ideally, if all the tables have similar data contents with an equal volume of DML activities, then the load can be easily split evenly across a given number of queues. In the real work environment, the tables in an IT system will have a wide variation in terms of datatypes, contents, volumes and work activities. Each table has its own characteristics that impose a level of impact on the replication’s queues. Bundling them up randomly in queues without due consideration will not only create unnecessary segregation of loads on certain queues but affect the overall replication throughput. Some of the queues with highly active and bigger load tables will take much longer for the backlogs to clear as compared to the other tables that have low volume and activities.
The challenge here is to approximate. For example, the global optimum arrangement of tables with different load characteristics to be serviced by a given number of queues so that the data replication process can clear the backlog in the replication in the least amount of time. Another method of solving an NP-hard problem is to try all the combinations available but that will be very expensive in terms of time and computational cost.

4.1 Estimating the cost of the solution

The solution in this paper is a configuration file that comprises of queues with tables allocated to them. The cost of a configuration-solution cannot be represented as a formula but through a series of application’s process against the Shareplex’s data replicating environment, the optimum configuration may not be able to yield the best throughput as the workload in any IT system’s databases in the real world tends to fluctuate from time to time. It is also difficult to predict the ability of the replication as there are many dynamic factors involved in the environment setup[3]. Our estimation attempt takes the following into consideration;

1) The list of tables and their data change activities for a defined period.
2) The allowable number of queues. An IT administrator can specify the max number of queues that can handle. Any excess number of queues above this threshold will be penalized but that shouldn’t stop the algorithm from considering them. If the additional queue(s) above the preference can bring in more benefits in comparison to the penalty it suffered, then it should be considered.
3) The measurement for a single queue performance will span from the source to the target databases. The weakness of a complete stack of replication’s setup lies with the weakness queues that are the most susceptible to experience backlogs the most.

![Algorithm 1 – SA for finding optimum Shareplex’s configuration](image)

Input: Shareplex configuration solution $S$, Length $L$
initialize1: Acceptance_probability $A$, random_number $r$, neighbouring solution of $S$ is $S'$
Initialize2: temp=1000, cooling constant=0.9, temp_min=1, old_cost=0
Initialize3: old solution cost cost_old, new_solution_cost cost_new
Result: approximate global optimum solution, $S_{opt}$

While temperature > temp_min do
  For $i$ = 1 to $L$ do
    #procedure to generate new configuration file
    $S' = sp\_generate\_new\_config$
    cost_old = cost($S$)
    cost_new = cost($S'$)
    if (cost_new - cost_old) <=0 then
      $S = S'$
    end
    if (cost_new - cost_old) >=0 then
      $A = e^{\frac{\text{cost\_new} - \text{cost\_old}}{\text{temperature}}}$
      $r = \text{random(1)}$
      #test probability to accept new solution
      If $A > r$ then
        $S = S'$
        cost_old= cost_new
      end
      $T = rT$  #Cooling temperature
  #procedure to restore all tables at source_site
  sp\_restore\_source\_tables
  #procedure to resync all tables at target site to source

The replicating process in a given queue begins at the source database; starting from the Data Capture process, then to the Read and followed by an Export process. The Export process transfer the information to the target DB’s Import process, which in turn divert to the Post process that converts the information into SQL statements and applies to the target DB. Each queue holds the information’s backlogs and has two statistics to show its progress activity: (1) the number of data statement changes; and (2) the time required to clear them. In the proposed approach, we sum up the statistics of each queue that determines its capability to handle the replicating load. The approach is to optimize the arrangement of the replicating tables across the most preferred number of queues, therefore, the cost of the configuration setting solution can be deemed as the summation of each queue that contains n numbers of tables, each with its own number of rows, row length, number of SQL updates and the time period that the load occurs. A cost factor is introduced to instill cost of exceeding the allowable number of queues allowed as in Eq(2):

\[
solution = \left( p \times \frac{1}{q} \sum_{n=1}^{m} (n_h \times n_k \times n_q) \right) t
\]  

(2)

Where \( n \) refers to the replicating tables, \( h \) is the number of rows that \( n \) has, \( j \) is the number of SQL workload activities are performed against the table, \( k \) is the size of a single row in the table, \( t \) is the time period, \( q \) is the number of queues and \( p \) is the cost of the queue maintenance. If it is less than the allowable threshold, the cost is 1. But if it exceeds the threshold, then the cost will be much higher as followed. This is to prevent the method from allocating an excessive number of queues that are beyond the allowable range as specified by the IT administrator. The proposed approach’s algorithm is listed in Algorithm 1.

5. Empirical Analysis

The experiments were conducted using two virtual machines that run on Linux OS and support two Oracle databases with Shareplex configured to replicate tables from a source DB to another. Each virtual machine has 4 GB of memory. The DB tables belong to a common DB schema and they comprised of a variety of data types; numeric, integer and varchar. We have created procedures that simulate a series of DML activities of these tables. The activities can be segregated into three groups; low, medium and high. Tables with low change activities receive less than 100 DML statements whereas those with high activities will get > 10000. The DML activities comprised of a mixture of delete, insert and update statements, all of which will be made on the tables at the source DB. The DML changes will then be propagated and applied to the target DB based on the Shareplex’s configuration file. The test setup is a controlled environment with no other IT application running against them. The maximum number of queues allowed is 5 for this test. 10 tables have been set up to replicate from the Source to the Target DB and a load test procedure was written to simulate the DML activity against the 10 tables with 100, 2500, 5000, 10000, 12500, 15000, 17500, 20000, 30000 and 40000 iterations.
For the simulated annealing algorithm, the control parameter is set with the bigger starting temperature and a smaller cooling which have a better chance of finding the optimal solution but require more iteration and time to execute. The final control parameters are used are as followed: starting temperature = 1000; final temperature = 1; and cooling rate – 0.9. Random solutions are generated throughout the process in every iteration under each temperate diminishing cycle. The randomly generated solutions’ costs are tracked in figure 4 and are measured in seconds. Their values are relative to the number of minutes in a day which is 1440.

![Fig. 4 – Generated random solutions’ costs](image1)

![Fig. 5 – SA’s selected solutions’ costs](image2)

Figure 5 showed the initial randomness in the selection of the solutions based on the SA’ algorithm in cycle 1. It started to converge at the 73rd iteration in cycle 1 onward. From the 2nd cycle onward, the results remain the same, adhering to the optimum solution with a cost of 0.085. This test has been repeated and the pattern of converging is the same which occurs during the 1st cycle while the other plateaued at the discovered optimum solution. Figure 6 showed the results from the five trial runs with the first cycle result shown. While the optimum configuration is the same discovered from the five test runs, their completion time has some slight variation, and this could be impacted by the OS environment against the Virtual Machines that the experiment is running on.

![Fig. 6– Results from Simulated Annealing (SA)](image3)

![Fig. 7 – Results from Hill Climbing (HC)](image4)

For the five test runs, the constraints set are on the number of iterations and queues available to support the table replication. As shown in figure 6, at the beginning where all the replicating tables are assigned to a single queue, the observed cost for the solution was above 80+ value, as the iteration proceeds, the proposed approach accepts solutions with a higher cost than the initials which is expected. As the temperature energy starts to cool down in SA, the tolerance level and probability threshold to accept worse off solutions is getting lower. During the initial half of the iterations, a higher level of fluctuation is seen among cost solutions that were accepted. But toward the end of the
run, the solutions with the lower cost have less volatility and remain around 63+ values. In Table 2, the results that were derived using the simulated annealing and hill-climbing heuristics. Compared with Figure 7 where the tests are conducted with a similar setup but with the Hill Climbing heuristics[24], the latter method only accepts better solutions’ costs and converges to solutions that are deemed as local optima. The results discovered by both heuristics can’t converge to the same cost for the different test runs. The reason is that they can approximate the optimums and the test setup could have minor fluctuations workloads and operating system activities that could influence the overall controlled environment.

The next test is to compare the difference in the data replications’ throughput between the optimum configuration file that is discovered by the SA method from the various test runs, versus another one that has been randomly generated. We subject the test under a variety of DB workload which runs a series of SQL updates to the tables and observes the time taken to clear the replicating backlogs, starting with 10 tables in replications for the 1st run. The subsequent test runs will have an increment of 10 more participating tables in replication and receive more updates. This repeats until the test reaches 50 tables with 550,000 updates in total. Figures 8 and 9 showed the results of the data replication tools’ throughput performance using both optimum and randomly generated configuration files against different SQL workloads, noting the big performance gap between the two as the workload increases.

**Table 2 – Throughput results from optimum configuration found by SA and HC heuristics**

| Runs | Throughput (sec) by SA | Number of Queues derived by SA | Throughput (sec) by HC | Number of Queues derived by HC | Tables’ workload |
|------|------------------------|-------------------------------|------------------------|-------------------------------|-----------------|
| 1    | 64.26                  | 5                             | 68.6                   | 5                             | 1000            |
| 2    | 65.29                  | 5                             | 69.1                   | 5                             | 1000            |
| 3    | 66.16                  | 5                             | 73.6                   | 5                             | 1000            |
| 4    | 67.46                  | 5                             | 79.3                   | 5                             | 1000            |
| 5    | 69.36                  | 5                             | 73.6                   | 5                             | 1000            |
| Average | 66.50                  | 5                             | 72.84                  | 5                             | 1000            |

It is evident that under a smaller load, there is very little difference in performance in the data replication setup between using the optimum and non-optimum configuration files. However, this difference becomes more evident when the loads exceed 55000 rows of changes. Overall, the optimum configuration can achieve 20+% better performance over the non-optimum ones.

**Fig. 8** – Performance results using optimum and non-optimum configuration setting for 5 queues

**Fig. 9** – Performance results using optimum and non-optimum configuration setting for 10 queues
6. Conclusion

A simulated annealing-based method has been proposed to find the optimum arrangement of tables for a pre-determined number of queues for the data replication setup that consists of databases and data replicating software. While the software used in our setup is primarily Oracle databases and Shareplex, the concept of using the algorithm to find the optimum configuration for the data replication software is generic and applicable to others. The simulation results demonstrate the effectiveness and efficiency of the proposed SA method. In the current context of data replication, the method to improve their throughput has been largely based on a series of vendors’ best practices and IT administrators’ experiences.

References

1. King, E., Automated Database Refresh in Very Large and Highly Replicated Environments. 2011.
2. Simitsios, A., P. Vasiliadis, and T. Sellis. Optimizing ETL processes in data warehouses. in null. 2005. IEEE.
3. Software, Q., Shareplex 9.0 Reference Guide. 2018.
4. Gupta, R., Introduction to Oracle GoldenGate (OGG), in Mastering Oracle GoldenGate. 2016. Springer. p. 3-10.
5. Gill, N.K. and S. Singh, A dynamic, cost-aware, optimized data replication strategy for heterogeneous cloud data centers. Future Generation Computer Systems, 2016. 65: p. 10-32.
6. Chapard, B. and M. Tomassini, Simulated Annealing, in An Introduction to Metaheuristics for Optimization. 2018. Springer. p. 59-79.
7. Software, Q., SharePlex 9.0 - Reference Guide. Quest Support, 2018.
8. Souravas, S. and A. Sifaleras, Trends in data replication strategies: A survey. International Journal of Parallel, Emergent and Distributed Systems, 2017: p. 1-18.
9. Hamdani, C., T. Hamrouni, and F.B. Charrada, Evaluation of site availability exploitation towards performance optimization in data grids. Cluster Computing, 2018. 21(4): p. 1967-1980.
10. Nazir, B., et al., The Impact of the Implementation Cost of Replication in Data Grid Job Scheduling. Mathematical and Computational Applications, 2018. 23(2): p. 28.
11. Blum, C. and A. Roli, Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM computing surveys (CSUR), 2003. 35(3): p. 268-308.
12. Assadi, M.T. and M. Bagheri, Differential evolution and Population-based simulated annealing for truck scheduling problem in multiple door cross-docking systems. Computers & Industrial Engineering, 2016. 96: p. 140-161.
13. Samora, I., et al., Simulated annealing in optimization of energy production in a water supply network. Water resources management, 2016. 30(4): p. 1533-1547.
14. Zaretalab, A., et al., A knowledge-based archive multi-objective simulated annealing algorithm to optimize series-parallel system with choice of redundancy strategies. Computers & Industrial Engineering, 2015. 80: p. 33-44.
15. Connell, A.M., An analysis of database replication technologies with regard to Deep Space Network application requirements. 2011.
16. Bahar, A., Use Case to S/4HANA Smart Data Integration (SDI), SAP, Editor. 2018.
17. Software, Q., Shareplex for Oracle v9.1.4. 2018.
18. Brunt, B., Going for gold: Dell Software's SharePlex database replication offering is a powerful tool with a small footprint. Computer Reseller News (UK), 2016: p. 23.
19. Dell Software Extends SharePlex to Optimize Data Integration and Analysis. Information Technology Newsweekly, 2013: p. 136.
20. Quest Software Releases SharePlex v9, in ICT Monitor Worldwide U6 - ctx_ver=Z39.88- 2004&ctx_enc=info%3Aofi%2Ffmt%3Atext%2FHTML& rft_id=info%3Aaid%2Fummon.serialsolutions.com&rft_val_fmt=info%3Aop%2Ffmt%3Akev%3Ainfo%3Aa%3Amed%3Ajournal&rft.genre=article&rft.atitle=Quest Software Releases SharePlex v9.rft.jtitle=ICT+Monitor+Worldwide&rft.date=2017-06-22&rft.pub=SyndGate+rMedia+Inc&paranddict=wen-US U7 - Newspaper Article. 2017. SyndGate Media Inc: Amman.
21. Nikolaev, A.G. and S.H. Jacobson, Simulated annealing, in Handbook of metaheuristics. 2010. Springer. p. 1-39.
22. software, Q., SharePlex 9.0 - Administration Guide. 2019.
23. Milani, B.A. and N.I. Navimipour, A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions. Journal of Network and Computer Applications, 2016. 64: p. 229-238.
24. Al-Bitar, M.A., S$beta$ S$-Hill$ climbing: an exploratory local search. Neural Computing and Applications, 2017. 26(1): p. 153-168.