Energy Efficient VM Placement for Effective Resource Utilization using Modified Binary PSO

- **Problem Formulation**

Resource Wastage Modeling

The resource wastage at physical server *j* can be modeled as equation (3.1)

$$W_j = \frac{\left|T_j^c - T_j^m\right| + \varepsilon}{U_j^c + U_j^m} \tag{3.1}$$

Where Wj is resource wastage T_j^c and T_j^m are remaining CPU and memory wastage respectively in normalized form ,U_j^c and U_j^m are CPU and memory resource usage respectively in normalized form.

Power Consumption Modeling

Power consumption at j^{th} physical server can be formulated as in equation (3.2).

$$P_j = \begin{cases} \left(P_j^{busy} - P_j^{idle}\right) \times U_j^c + P_j^{idle}, & if\ U_j^c > 0 \\ 0 \ \ \text{otherwise} \end{cases} \tag{3.2}$$

Where P_j^{busy} and P_j^{idle} are the consumed power values when the j^{th} physical server is fully loaded and idle respectively. Based on the observations, obtained from the experimental work, these parameters, in our experiment, are fixed to 215 and 162 Watt respectively.

Model contd..

- **Notations used**

Notation	Meaning
W_j	Resource wastage at j^{th} physical server.
T_j^c, T_j^m	Remaining CPU and remaining memory wastage at j^{th} physical server.
U_j^c, U_j^m	Normalized usage of CPU and memory.
P_j^{busy}, P_j^{idle}	Power consumption at j^{th} physical server when j^{th} server is fully loaded and idle.
$I = \{1, 2, \ldots\ldots\ldots\ldots, n\}$	Set of 'n' VMs.
$J = \{1, 2, \ldots\ldots\ldots\ldots, m\}$	Set of 'm' physical servers.
ρ_{c_i}, ρ_{m_i}	CPU and memory requirement of VM i.
x_{ij}	A variable that indicates whether the i^{th} VM is assigned to j^{th} physical server or not.
y_j	A variable that indicates whether the j^{th} physical server is in use or not.
R_{c_j}, R_{m_j}	Thresholds of CPU utilization and memory utilization corresponding to j^{th} physical machine.
P_{known}	Evaluated Pareto front.

Model contd..

Mathematical Formulation

Let x_{ij} and y_j be two binary variables as follows.

$$x_{ij} = \begin{cases} 1 & \text{if the VM } i \in I \text{ is assigned to physical server } j \in J \\ 0 & \text{otherwise} \end{cases}$$

$$y_j = \begin{cases} 1 & \text{if the server } j \in J \text{ is in use} \\ 0 & \text{otherwise} \end{cases}$$

So the problem can be formulated as given in equations (3.3) and (3.4).

$$f_1 = Minimize \sum_{j=1}^{m} W_j = \sum_{j=1}^{m} \left[y_j \times \frac{\left| \left(R_{c_j} - \sum_{i=1}^{n} (x_{ij} \cdot \rho_{c_i}) \right) - \left(R_{m_j} - \sum_{i=1}^{n} (x_{ij} \cdot \rho_{m_i}) \right) \right| + \varepsilon}{\sum_{i=1}^{n} (x_{ij} \cdot \rho_{c_i}) + \sum_{i=1}^{n} (x_{ij} \cdot \rho_{m_i})} \right] \qquad (3.3)$$

$$f_2 = Minimize \sum_{j=1}^{m} P_j = \sum_{j=1}^{m} \left[y_j \times \left((P_j^{busy} - P_j^{idle}) \times \sum_{i=1}^{n} (x_{ij} \cdot \rho_{c_i}) + P_j^{idle} \right) \right] \qquad (3.4)$$

Model contd..

Mathematical Formulation

Subject to:

$$\sum_{j=1}^{m} x_{ij} = 1 \quad \forall i \in I \tag{3.5}$$

$$\sum_{i=1}^{n} \rho_{c_i} \cdot x_{ij} \le R_{c_j} \cdot y_j \qquad \forall j \in J \tag{3.6}$$

$$\sum_{i=1}^{n} \rho_{m_i} \cdot x_{ij} \le R_{m_j} \cdot y_j \qquad \forall j \in J \tag{3.7}$$

Model contd..

The Proposed VM Placement Method

An individual particle, in the swarm, is represented by a set of three vectors

$$\left\langle \vec{X}_i, \vec{P}_i, \vec{V}_i \right\rangle$$

Each of which is a *d* dimensional vector, where *d* is the cardinality of the search space dimension.

velocity of particle is updated after k^{th} iteration as in equation (3.8).

$$v_{id}^{k+1} = w \cdot v_{id}^{k} + c_1 \cdot \varphi_1 (\overrightarrow{pbest}_{id}^{k} - x_{id}^{k}) + c_2 \cdot \varphi_2 (\overrightarrow{gbest}_{d}^{k} - x_{id}^{k}) \qquad (3.8)$$

c_1 and c_2 are the cognition learning and social learning rate respectively. *w* is the inertia weight controlling the velocity. φ_1 and φ_2 are real random numbers in the range [0, 1].

Based on this velocity, the position of the particle is updated as in equation (3.9).

$$\mathrm{x}_{id}^{k+1} = \begin{cases} 1 \;\; if \left(\varphi < S(v_{id}^{k+1}) \right) \\ 0 \;\; otherwise \end{cases} \qquad (3.9)$$

Here, S is a sigmoid function and φ is quasi-random real number distributed uniformly in [0, 1] as defined in equation (3.10)

$$S(v_{id}^{k+1}) = \frac{1}{\left(1 + \exp(-v_{id}^{k+1})\right)}$$

Model contd..

Solution Representation

In BPSO, each x_{ij} is a decision variable which represents whether i^{th} VM is assigned to physical server j or not. Each solution or particle in the VM allocation problem is represented by a binary matrix as in Figure below, with the condition that row sum should be equal to one.

$$\begin{bmatrix} 0 & 1 & \dots 0 & \cdots & 0 \\ 1 & 0 & \dots 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & & 1 \\ \vdots & \vdots & \ddots & & \vdots \\ 1 & 0 & \dots 0 & \dots & 0 \end{bmatrix}$$

Model contd..

Particle Initialization

Algorithm 1: *Generator(n, m)*
1. $x = zeros(n, m);$ // Generates a matrix of n × m
2. $j = randi([1\ m],1,n);$ // Generates a vector of n×1 that has all values randomly between [1,m]
3. *for* $i = 1:n$
$x(i,\ j(i))=1;$ // Assigning a particular VM to a physical machine
4. *end*

Position Update of Particles

Model contd..

Selection of *$pbest_i$* and *gbest*

- With the help of dynamic neighborhood concept, the $pbest_i$ and gbest are calculated as follows.
- Calculate the distance between current particle and the other particle in fitness value space of the first objective function (f_1).
- Based on the above calculated distance find the nearest *l* particles as the neighbors of the current particle.
- Among these $l+1$ particles, the $pbest_i$ is calculated using the second fitness function (f_2).
- When any of the two, (f_1 and f_2), is lower than the current particle then update the location of $pbest_i$.
- If both of the f_1 and f_2 are lower than the current particle then the location of *gbest* is updated.

Model contd..

Pareto Set Calculation Procedure

Algorithm 2: *Calcultate_Pareto(S')*

Input: *Set of solutions S'*

Output: *Set of approximate Pareto optimal solutions P*

1. *Start with i=1.*

2. *Using the concept of dominance find $j \neq i$, such that $\vec{x}^{(j)} \preceq \vec{x}^{(i)}$.*

3. *If j is found then mark $\vec{x}^{(i)}$ as dominated solution and increment the value of i by one. Go to step 2.*

4. *If all the solutions in the set have been considered (i.e. when i = N) then go to step 5. Otherwise increment the value of i by one and go to step 2.*

5. *Return the set of solutions which are unmarked as a non-dominated solution set or approximate Pareto optimal set P.*

Model contd..

VMPBPSO Algorithm

Algorithm 3: VMPBPSO

Input: *Set of VMs with their computing and memory demands, Set of physical servers with their capacity threshold and set of parameters.*

Output: *An approximate Pareto optimal set P representing possible VM allocations.*

1. *Initialize the swarm of particles using generator function in the binary search space.*
2. *While stopping criteria is not met do*
3. *Compute the fitness of each particle using equation (3.3) and (3.4) and update the gbest and pbest.*
4. *For all particle do*
5. *Update the position and velocity using equations (3.9) and (3.8) respectively.*
6. *End for*
7. *End while*
8. *Calcutate the approximate pareto optimal set P from the obtained solution using Calcultate_Pareto().*

Model contd..

The flow chart of the proposed VMPBPSO

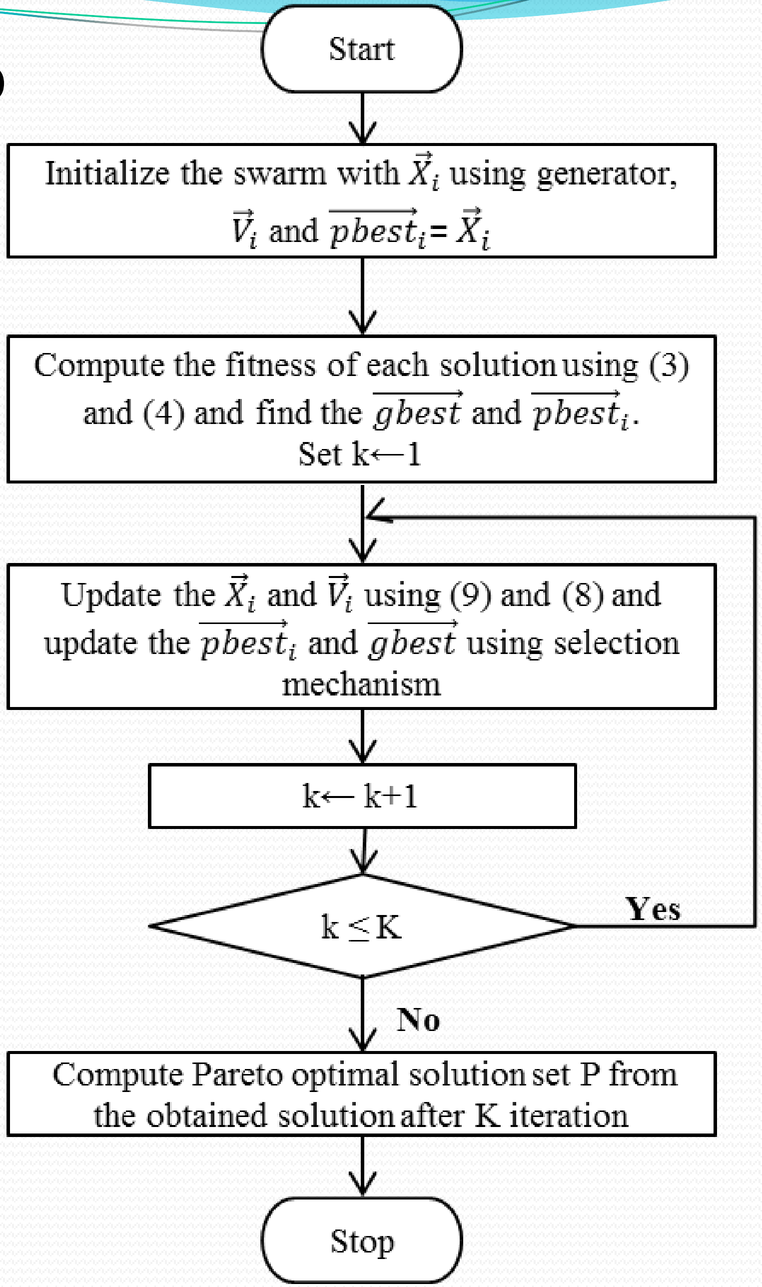


Figure 5.2: Flow chart

Model contd..

- **Experimental Analysis**

Table 5.1: Comparison between MGGA, VMPACS and VMPBPSO with respect to ONVG and spacing

Reference value	Correlation coefficient	Algorithm	ONVG	Spacing
$\bar{\rho}_c = \bar{\rho}_m = 25\%$	-0.754	MGGA	16.02	0.59
		VMPACS	21.24	0.21
		VMPBPSO	24.32	0.18
	-0.348	MGGA	17.08	0.52
		VMPACS	23.42	0.19
		VMPBPSO	25.11	0.15
	-0.072	MGGA	15.74	0.46
		VMPACS	18.65	0.15
		VMPBPSO	22.21	0.12
	0.371	MGGA	14.24	0.32
		VMPACS	19.14	0.14
		VMPBPSO	25.74	0.09
	0.755	MGGA	15.06	0.21
		VMPACS	24.27	0.12
		VMPBPSO	25.67	0.07
$\bar{\rho}_c = \bar{\rho}_m = 45\%$	-0.755	MGGA	16.38	0.22
		VMPACS	21.56	0.17
		VMPBPSO	23.51	0.15
	-0.374	MGGA	14.23	0.20
		VMPACS	22.40	0.16
		VMPBPSO	24.20	0.13
	-0.052	MGGA	13.45	0.19
		VMPACS	21.17	0.14
		VMPBPSO	25.47	0.10
	0.398	MGGA	11.98	0.16
		VMPACS	20.45	0.11
		VMPBPSO	27.84	0.08
	0.751	MGGA	10.54	0.13
		VMPACS	18.64	0.08
		VMPBPSO	29.36	0.04

Model contd..

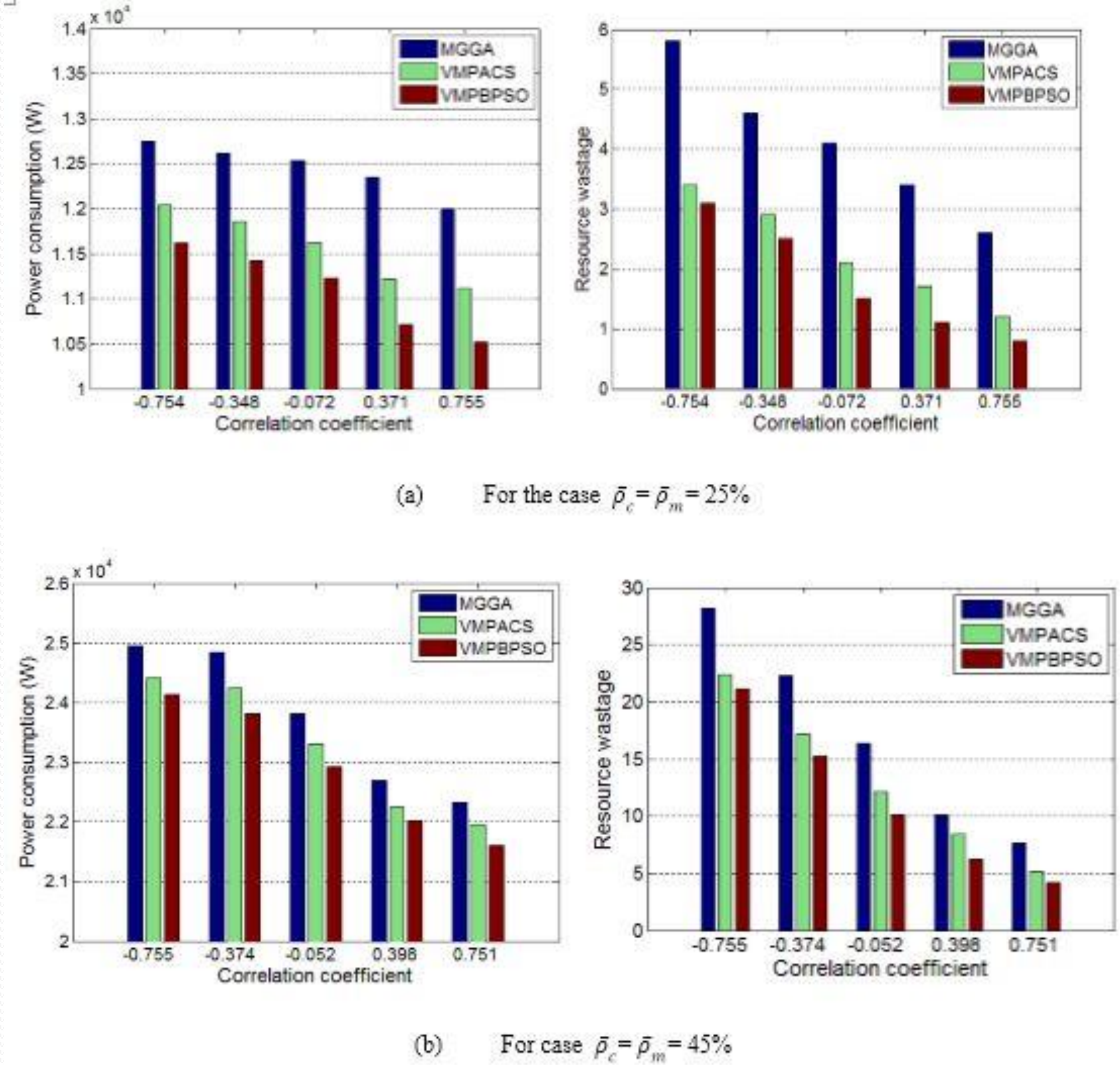
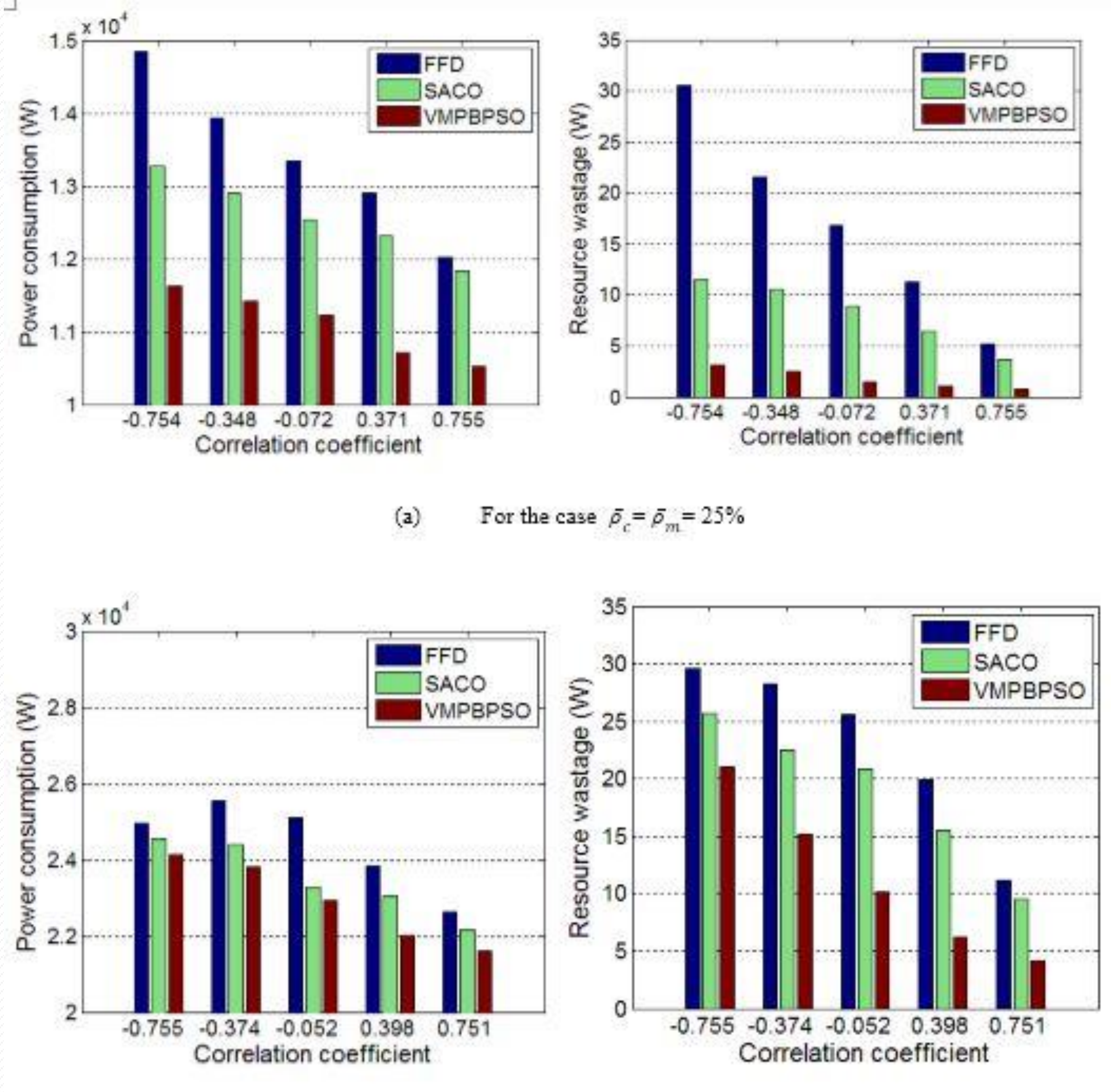


Figure 5.3: Comparison with VMPACS and MGGA for Power Consumption and Resource Wastage

Model contd..

Figure 5. 6: Comparison of VMPBPSO with FFD and SACO

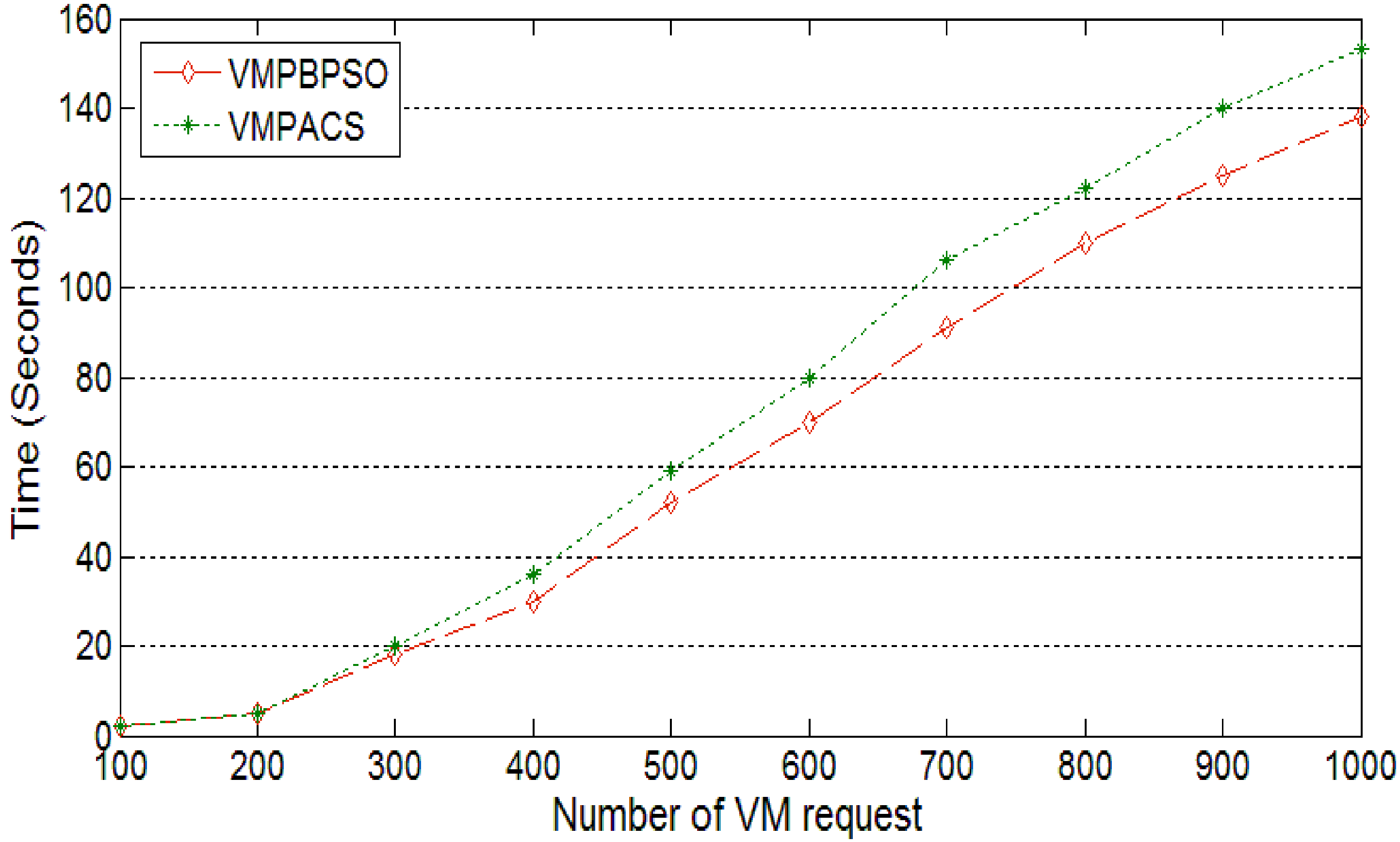


(a) For the case $\bar{\rho}_c = \bar{\rho}_m = 25\%$

(b) For case $\bar{\rho}_c = \bar{\rho}_m = 45\%$

Model contd..

Figure 5.7: Convergence of VMBPSO



Model contd..

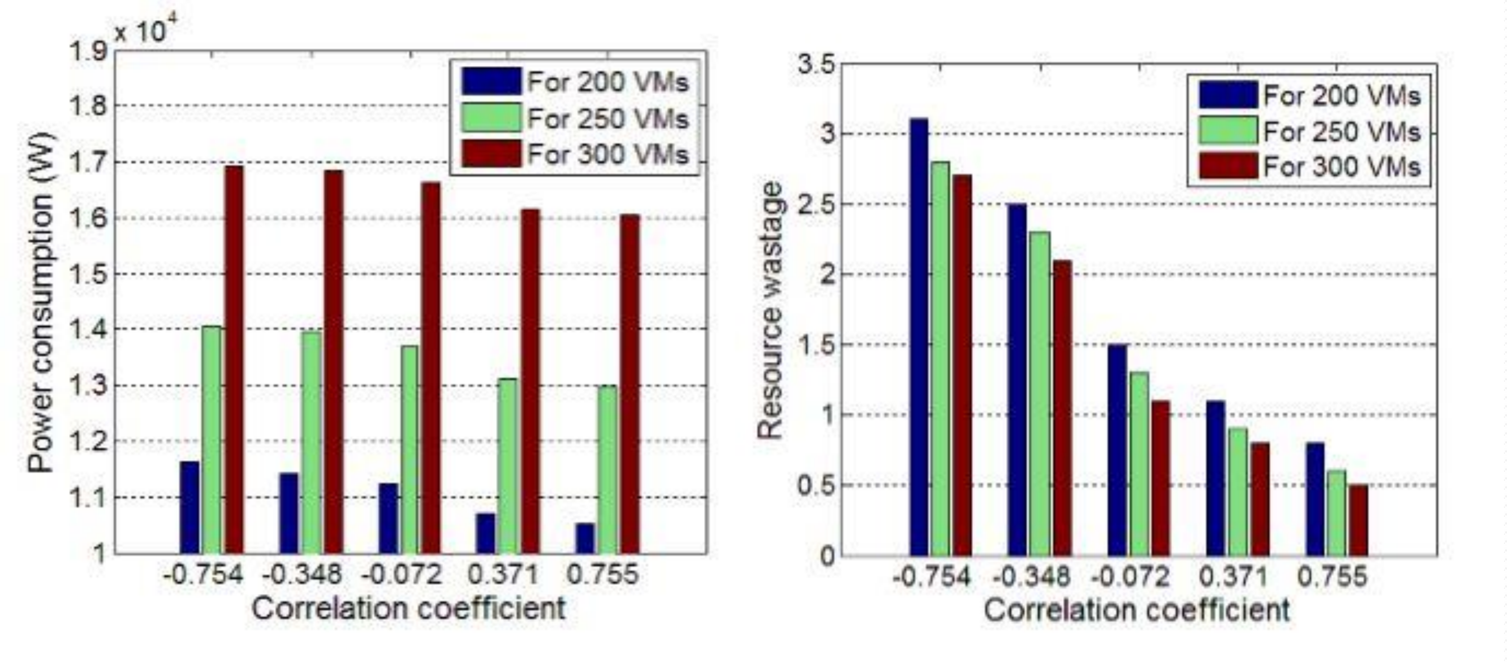
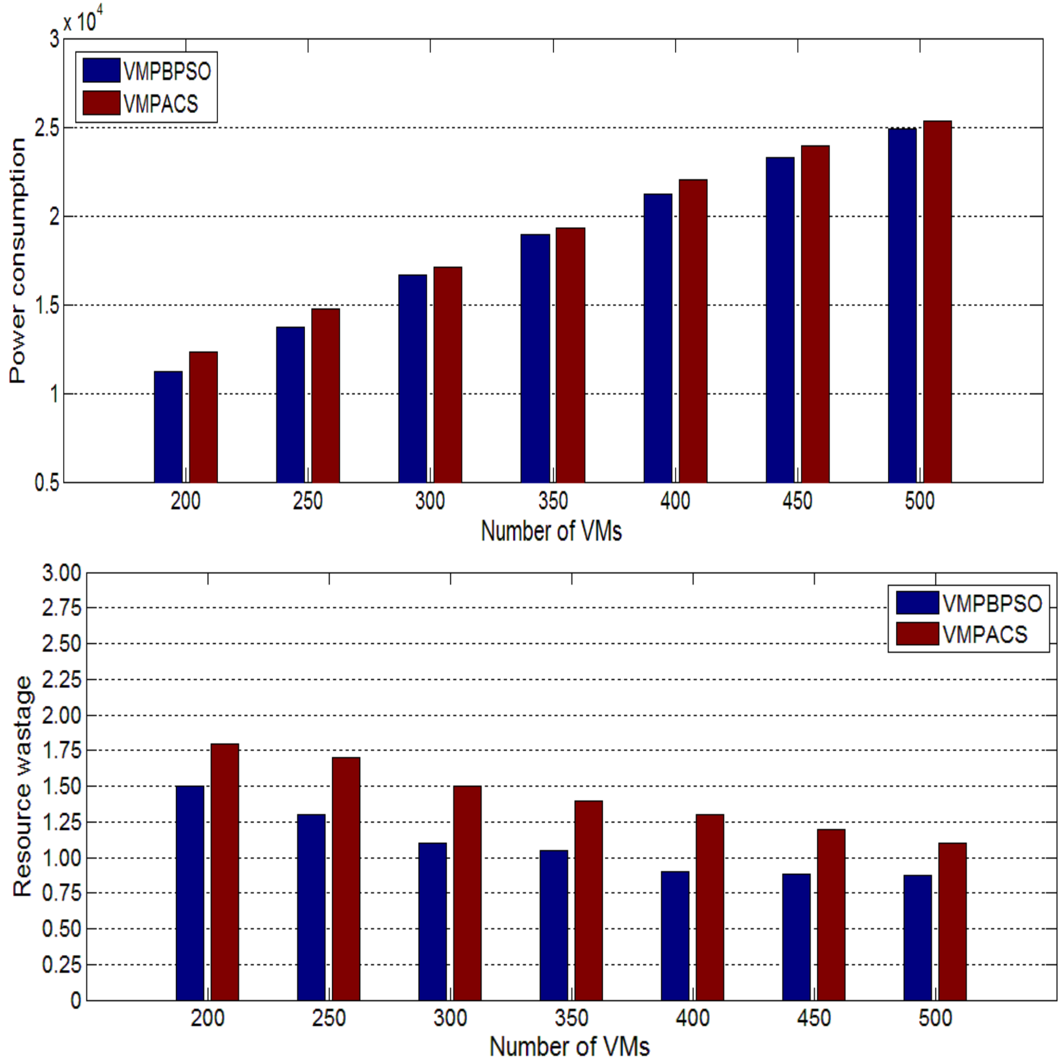


Figure 5.8: Power consumption and resource wastage for various VMs

Model contd..

Figure 5.9: Performance on Large Number of VMs



Model contd..

Conclusion & Future Work

- The proposed VMPBPSO algorithm performs better on these two objectives as compared to contemporary algorithms VMPACS, MGGA, SACO and FFD for the same problem.
- VMPBPSO employs less physical servers for the placement of virtual machines and also explore the search space efficiently resulting in better performance.
- The obtained results establishes that the concept of dynamic neighborhood favors less number of physical servers therefore overall performance of the proposed method is improved.
- The limitation of the proposed method is that this work does not consider the dynamic nature of the VM placement.
- The future work will establish to formulate hybrid meta-heuristic techniques implied to solve the VM placement problem for better possible solutions.
- It is also possible to consider the network resources such as network bandwidth for VM placement in addition to computation and memory resources.

Questions & Suggestions?