Journal of Economics and Administrative Sciences Vol.27 (NO. 129) 2021, pp. 154-170

Journal of Economics and Administrative Sciences (JEAS)

Available online at http://jeasiq.uobaghdad.edu.iq

Solving Resource Allocation Model by Using Dynamic Optimization Technique for Al-Raji Group Companies for Soft Drinks and Juices

Mohammed Hasan
College of Administration and Economics /University of Baghdad/Iraq
Mohammed.hasan1101a@coadec.uobaghdad.edu.iq

Mohammed Sadiq Al-Doori
College of Administration and Economics /University of Baghdad/Iraq
dr_aldouri@coadec.uobaghdad.edu.iq

Received:22/3/2021 Accepted: 10/5/2021 Published: SEPTEMBER/ 2021

Abstract:

In this paper, the problem of resource allocation at Al-Raji Company for soft drinks and juices was studied. The company produces several types of tasks to produce juices and soft drinks, which need machines to accomplish these tasks, as it has 6 machines that want to allocate to 4 different tasks to accomplish these tasks. The machines assigned to each task are subject to failure, as these machines are repaired to participate again in the production process. From past records of the company, the probability of failure machines at each task was calculated depending on company data information. Also, the time required for each machine to complete each task was recorded. The aim of this paper is to determine the minimum expected time for the completion of all the machines assigned to perform their tasks in the company by using the dynamic optimization technique over finite and infinite horizons. By comparing the results, it was found that the first and second tasks were better than the third and the fourth tasks because the first task and the second one had completed their tasks in a shorter period than the others, they took 1379.2 hours and 1379.3 respectively during $H = \infty$ of horizons (stages), while the third task took 1379.4 hours and the fourth task 1379.5 hours. A careful analysis of the situation revealed that the time it takes for each machine to complete its tasks has been reduced, from appropriate planning and quick and effective maintenance can enhance the capacity of the machines and thus reduce time and effort, which contributes to reducing the company's costs and thus maximizing the production capability to increase the company's profits.

Keywords: **Resource Allocation Problem, Dynamic Optimization, Finite and Infinite Horizon, Value Iteration Algorithm.**

1. Introduction:

There are many methods for solving mathematical models that are mainly designed to solving the problems and reach the optimal solution for a specific mathematical model, and they are used in many different fields, one these fields is Operation Research. In this section, we are defining our approach to build the model of resource allocation and solving it, using dynamic optimization technique and explaining the problem. There are many models in Operation Research such as (Resource Allocation, Game theory, et al...) which are solved by using many techniques and one of these techniques is Dynamic Programming.

Dynamic programming is a mathematical technique for solving problems that seeks the optimal solution for a multi- stage problems. The optimal solution may mean maximizing profits, minimizing costs or time to complete the task, which had to do with finding optimal decisions. Dynamic programming is used to create a series of interconnected decisions by diving the problem into several stages, in each stage there are many decisions and the best one is chosen. In order to solve any problem, it needs a decision to make, in which the time factor plays the primary role, and therefor the decision is made in multiple stages, where the problem is divided into multiple parts and each part represents a stage, and the decision is made at the end of each stage. Every decision that is made depends on the previous stage and affects the next stage to which it is related. This study includes solving the Resource Allocation problem using the dynamic optimization technique with using value iteration algorithm to find the optimal solution. We formulate the model and consider to allocate different machines to many different tasks in a production company. The production company have a limit number of machines and wants to allocate them to the tasks at the beginning of planning horizon. Where the company estimated the working times of each machines and their distribution to accomplish each task to be received in the first stage of the process. Also the probability of failure of each machine assigned to each task was calculated. The machines that got broken are to be repaired to participate in the production process, as it will be allocated at the beginning of the next period.

The problem is formulated as a dynamic optimization problem and based on previous studies. In (1969) Nemhauser & Ulmann give a study considering the discrete dynamic programming and capital allocation, where dynamic programming algorithms have been developed for optimal capital allocation according to budgetary constraints [7]. Abraham & Paul published a paper in (1985) that was about generalized polynomial approximations in Markovian decision processes; through this study they found that the structure of the value function in the process of making a Markovian decision through the linear superposition of the base function reduces the dimensional problem [18]. Mulvey & Valdimirou published a paper in (1992) and they discussed stochastic network programming for several financial planning problems that are posed as dynamic programming with stochastic parameters [13]. And in (1997) Bertsekas & et al. discussed an application of neuro-dynamic programming technique to the optimization of retailer inventory system they compared the performance of solutions generated by using Neuro-dynamic programming algorithms [27].

In (2002) Godfrey & Powell proposed a study that addressed an adaptive dynamic programming algorithm for dynamic fleet management, I: Single period travel times. They considered that the problem is a random version of the dynamic resource allocation problem, and they were solved the problem by using an adaptive dynamic programming algorithm that uses nonlinear function approximations that yield resource values in the future [8]. Van Roy & Farias published a paper in (2003) about the linear programming approach to approximate dynamic programming; they developed error limits that provide performance guarantees and also guide the selection of each of the fundamental functions that affect approximation quality [4]. Then in (2003) Guestrin & et al.'s paper was efficient algorithms for factored MDPs; the paper addressed the problem of planning under uncertainty in large Markov Decision Processes they provide experimental results on problems with large state space [9]. In (2004), Powell & Spivey studied the dynamic assignment problem; they proposed a hierarchical aggregation strategy for problems where the attributes spaces are too large to be enumerated [24]. In (2005) Osman & el al. proposed a study that deal with an effective genetic algorithm approach to multi-objective resource allocation problems (MORAPs); through this study, they used genetic algorithm to solve a multi-objective resource allocation problems. Also, this approach is developed to deal with both multi-objective problems and single problems [17]. In (2005), Powell & et al. published a paper about approximate dynamic programming for high dimensional resource allocation problems [22]. Topaloglu & Kunnumkal in (2006) published a paper about approximate dynamic programming methods for an inventory allocation problem under uncertainty [25]. After that, Samuel & et al. presented general method to solve infinite-time horizon problems in (2006) through the exhaustive description of an algorithm that they implement to determine the optimal strategy. From this study, their model yields two predictions. First, spiders reduce their web size as they are gaining weight due to body mass dependent cost of web building. Second, this reduction in web size starts at lower weight under higher ferocity risk [28]. Then, Powell & Topaloglu in (2006) proposed a study about Dynamic-programming approximations for stochastic time-stage integer multi-commodity flow problems [26]. Then in (2012), Nwozo & Nkeki published a paper about a dynamic optimization technique for resource allocation problems in a Production Company [16]. Noor & Doucette published a paper in (2012) about an applications of infinite horizon stochastic dynamic programming in multi-stage project investment decision-making [15]. Later, Ajofoyinbo & Orolu made a study in (2012) about optimal allocation of radio resource in cellular LTE downlink based on truncated dynamic programming under uncertainty [1]. Then came Hauskrecht & Singliar in (2012) to make a study had concern Monte-Carlo optimizations for resource allocation problems in stochastic network systems [10]. Then in (2013),Nkeki offered a study in connection with Dynamic optimization technique for distribution of Goods with Stochastic shortage [14]. A paper was published in (2017) by Amuji & et al.; they made a case study in Nigerian university and the paper's address was the usefulness of dynamic programming in course allocation in the Nigerian university [3].

After that, Hoang & et al. published a paper in (2017) about Infinite-horizon proactive dynamic DCOPs [12]. In (2017), Zhongkai & et al. made a study about dynamic optimization Two Dimension Reduction Methods for Multi-Dimensional Dynamic Programming and Its Application in Cascade Reservoirs Operation Optimization and they used two methods to solve Multi-dimensional dynamic programming in china [29]. This paper is organized and detailed in the following parts. Section (2) will describe the problem and our target of using this type of methods. In Section (3), the probabilistic dynamic programming will be defined. The formulation of the resource allocation problem and defining the notations will be explained in section (4). While in section (5), the computational results will be discussed. And conclusion follows in section (6).

2. Problem description:

While there are many shapes that used in allocation decision problem can take. This paper describes one type of these shapes and will describe it in a more high-level form to set stage. A company has a number of tasks and wants to allocate machines at each. These machines are expected to get failed due to the uncertainty of the number of machines that fail, then we suppose that the states are random; the company should know the number of machines that would be available for the next period and before making a decision on how to allocate the remaining machines for each task. The number of machines to be participate into operation in the next period depends on the number of machines that failed at the end of previous period. The aim of this paper is to determine minimum expected time for the completion of all the machines that assigned to perform their tasks in the company by using dynamic optimization technique which is the optimal solution at each stage and then link all solutions to reach a final solution of the problem over H horizon of study.

3. Probabilistic Dynamic Programming:

In the previous section, the dynamic programming is defined as a mathematical method that aims find the optimal solution to a specific problem to make an interrelated sequence of decisions after dividing the problem into many parts, where each part represents as a stage and linking these stages with each other by a mathematical relationship, and each stage of the problem contains a set of states.

Dynamic programming is used to solve many cases, whether these cases are deterministic or non-deterministic. Probabilistic dynamic programming differs from the deterministic in that the states in the next stage is indefinite and known from the states in the current stage, where the states in this model are random. Therefor, this type of dynamic programming is more difficult and complicated than the deterministic type [11].

The dynamic programming model can be considered as a probabilistic model if the following conditions are met:

- **If the contributions that are related to the states are uncertain on unknown.**
- **If the resulting states in each of the stages are random and uncertain.**

when the above conditions are available, this means that the system can be defined under uncertainty, and in this case the expectation taken to find the optimal solution for the model; equation (1) is called as the recurrence equation or Bellman's equation.

$$F_h(s_h) = \underset{x_h}{\text{opt}}\, E\{\Omega_h(S_h, x_h) + F_{h+1}(s_{h+1})\} \quad (1)$$

Where,

$\Omega_h(S_h, a_h)$: **The returns obtained are represented by state (S) when event (a) occurs at stage (h).**

$F_h(s_h)$: **The value function of state (s) at stage (h) [19].**

4. Formulate the Allocation Problem and Optimality Equation:

The allocation problem requires the existence of sources and receivers. Sources mean the parts that need to allocate to a recipient or an outfall, and it is represented by the allocation of machines, employees, or … etc. As for the recipients, it means the task to which the source is allocated. We have taken into consideration in the study of this problem, which is the allocation of a certain number of machines to a certain number of tasks at the beginning of each period of time. The assigned machines for each task are subject to failure. Thus, we expect some machines to breakdown at the end of each period, which leads to the uncertainty in the number of machines that will be failed at the end of the period. For this reason the states of the machines are random. The production company must know the number of machines that will be available in the next period before it makes any decision regarding assigning tasks to the functioning machines. The number of machines that will participate in the production process for the next period depends on the number of machines that failed at the end of each previous period. The goal is to determine the expected times for each machine to complete each task assigned during certain time periods. If the time accomplished for the machines, which resulted from allocating m of tasks to the machines during a period h is Ω_h^k, and the states of machines is S_h. Then, the number of machines assigned to the production process for each task in each period is according to the policy π is $x_h^{\pi^k}$, and the number of machines that failed is b_h, the returns that will be achieved for the company over H horizons is given by:

$$\sum_{h=0}^{H}\sum_{k=1}^{m} \Omega_h^k\left(x_h^{\pi^k}(s_{h-1}(b))\right)$$

The minimum expected time accomplished for the company's machines under the policy π as in the following equation:

$$Y_h^{\pi}(s_h) = E\left[\min_{x_h \in X(s_h)} \sum_{h=0}^{H}\sum_{k=1}^{m} \beta^h \Omega_h^k\left(x_h^{\pi^k}(s_{h-1}(b))\right)\right] \quad (2)$$

Subject to

$$\sum_{k=1}^{m} x_h^{\pi^k}(s_{h-1}(b)) \geq s_h, h = 0, 1, 2, \dots, H$$

$$x_h^{\pi^k}(s_{h-1}(b)) \geq 0, h = 0, 1, \dots, H; k = 1, 2, \dots, m$$

Where

β: **is the discount factor $0 < \beta < 1$.**

S: the state space *i.e.* the set of all machines.
H: the set of stages in the planning Horizon.
π: the policy of the system which is the rule of an action chosen based on current state.
$x_h^{\pi^k}$: Number of machines that allocated to task k at stage h under policy π, $h \in [0, H]$.
s_h: Number of available machines at period h; $h \in [0, H], s_h \in S$.
b: Number of machines who failed at period h.
$x_h^k(s_{h-1}(b))$: Number of available machines to be allocated in the next period.
$X(s_h)$: It represents the set of possible solutions to the problem, $x \in X$.
Ω_h^k: The expected returns (contributions) from task k at stage h, $h \in [0, H]$.
Π: the set of all policies, $\pi \in \Pi$.
s_0: The initial number of machines available at the beginning of planning horizon.
$Y_h(s_h)$: The objective function, $h \in [0, H]$.
Problem (3) minimizing the expected times over $X(s_h)$

$$Y_h^{\pi}(s_h) = E\left\{\min_{x^{\pi}\epsilon X(s)} \sum_{h'=h}^{H} \sum_{k=1}^{m} \beta^h \Omega_{h'}^k \left(x_{h'}^k(s_{h'-1}(b))\right) | s_h \in S_h\right\} \quad (3)$$

Subject to

$$\sum_{k=1}^{m} x_h^{\pi^k}(s_{h-1}(b)) \geq s_h, h = 0, 1, 2, \dots, H$$

$$x_h^{\pi^k}(s_{h-1}(b)) \geq 0, h = 0, 1, \dots, H; k = 1, 2, \dots, m$$

If we accumulate the returns of the first H-stage and add to it terminal returns.

$$\Omega_H(s_H) = \sum_{k=1}^{m} \Omega_H^k(s_H)$$

Then (3) will be

$$Y_h^{\pi}(s_h) = E\left[\left\{\min_{x^{\pi}\epsilon X(s)} \sum_{h'=h}^{H-1} \sum_{k=1}^{m} \beta^h \Omega_{h'}^k \left(x_{h'}^k(s_{h'-1}(b))\right)\right\} + \beta^H \Omega_H(s_h) | s_h \in S_h\right] \quad (4)$$

Subject to

$$\sum_{k=1}^{m} x_h^{\pi^k}(s_{h-1}(b)) \geq s_h, h = 1, 2, \dots, H$$

$$x_h^{\pi^k}(s_{h-1}(b)) \geq 0, h = 0, 1, \dots, H; k = 1, 2, \dots, m$$

Since the states (s) are random, by using s_h as the state variable at stage h and S as the state space. We can formulate this problem as a dynamic programming, the number failed machines for task k at stage h is $R_k x_h^k$, where R_k the probability of failure machines at each task k. The total number of failure machines for all tasks is as given:

$$b = \sum_{k=1}^{m} R_k x_h^k \quad , \quad h = 1,2,\dots,H$$

Assume that s_h be the number of machines to be allocated at stage h, and α be the percentage of breakdown machines that are expected to join the functional machines in the next period [16].

$$s_h = s_{h-1} - \left((1-\alpha) * \sum_{k=1}^{m} R_k x_h^k \quad , h = 1,2,\dots,H \right)$$

By substitute β in place of $(1-\alpha)$ then the equation will becomes.

$$s_h = s_{h-1} - \beta * \sum_{k=1}^{m} R_k x_h^k \quad , \quad h = 1,2,\dots,H \qquad (5)$$

And using equation (6) we can compute R.

$$\hat{R}(i) = 1 - \frac{i}{n+1} = \frac{n+1-i}{n+1} \qquad (6)$$

For more details see [5], after that we used the Reliability of the system [6].

$$R(s) = \prod_{i=1}^{n} R(i) \qquad (7)$$

In this case we can find the optimal policy and computing the value function through the optimization problem which is given by

$$Y_h^{\pi}(s_h) = \min_{x^{\pi} \in X(s_h)} \left[\sum_{k=1}^{m} \Omega_h^k \left(x_h^{\pi^k}(s_{h-1}(b)) \right) + (E\{F_h(s_h)\} | S_h = s_h) \right] \qquad (8)$$

Subject to

$$\sum_{k=1}^{m} x_h^{\pi^k}(s_{h-1}(b)) \geq s_h, \quad h = 1,2,\dots,H$$

And equivalently

$$\sum_{k=1}^{m} x_h^{\pi^k}(s_{h-1}(b)) = s_h + \xi_h \quad , h = 1,2,\dots,H$$

$$x_h^{\pi^k}(s_{h-1}(b)) \geq 0, \quad h = 0,1,\dots,H; \; k = 1,2,\dots,m$$

Since all the working and available machines to be allocated in the next stage are less than the machines that must be allocated to reduce the time required to complete each machines there each task, in addition we have $\xi_h = 0$, $\forall \, h \in H$.

***Lemma 4-1*: let s_h be a state variable and let $Y_h(s_{h+1})$ be a function who measured at some point $h' \geq h+1$ conditional the random variable s_h. Then**

$$E\{E(F_{h'} | S_{h+1}) | S_h\} = E\{F_{h'} | S_h\} \qquad (9)$$

By this lemma and for more detail about the proof see [20].

***Theorem 4-1-1*: now we suppose that $Y_h^{\pi}(s_h)$ is a solution of (7), then $Y_h^{\pi}(s_h) = F_h^{\pi}(s_h)$ and to show that at first, we use a standard dynamic programming through this relation**

$Y_H^{\pi}(s_H) = F_H^{\pi}(s_H) = \Omega_H(s_H)$ And it holds for *h+1, h+2… H* to see details [20]. That will leads to:

$$F_h(s_h) = \min_{x_h \in X(s_h)} \left[\sum_{k=1}^{m} \Omega_h^k \left(x_h^k(s_{h+1}(b)) \right) + E\{F_{h+1}(s_h)\} \right] = Y_h^\pi(s_h)$$

And

$$F_h^*(s_h) = \min_{\pi \in \Pi} F_h^\pi(s_h)$$

Then the optimality equation becomes see [16]:

$$F_h(s_h) = \min_{x_h \in X(s_h)} \left[\sum_{k=1}^{m} \Omega_h^k \left(x_h^k(s_{h+1}(b)) \right) + F_{h+1}(s_{h-1} - \beta \sum_{k=1}^{m} R_k x_h^k) \right] \qquad (10)$$

Now by using the equation (10) to show that for any $\varepsilon > 0$, exists $\pi \in \Pi$ And satisfies the condition:

$$Y_h^\pi(s_h) + (H - h)\varepsilon \geq F_h(s_h) \qquad (11)$$

The result of equation (11) proves that solving the optimality equation also gives the optimal value function [16].

To solve equation (10), we used the value iteration algorithm for finite and infinite stage by using forward dynamic programming [28], and as follows:

Step 1: Initialization

$$h = 0, \forall\, s \in S, F_0(S) = 0, \beta = 0.05, \varepsilon = 0.001$$

Step 2: Iterative process

$h = h + 1,$

For each $s \in S$, $f = F_h(s)$

For each $x \in X$, calculate:

$$F_{h+1}(s_h) = \min_{x_h \in X(s_h)} \left[\sum_{k=1}^{m} \Omega_h^k \left(x_h^k(s_h(b)) \right) + F_h(s_{h-1} - \beta \sum_{k=1}^{m} R_k x_h^k) \right]$$

$\pi_h(s) = arg\ min\ F_h(s, x)$

$F_h(s) = F_h(s, \pi_h(s))$

Step 3: if $max\ (|F_h(s) - f|) \leq \varepsilon$

$\pi_h(s) = \pi^*$, We found the optimal strategy and stop. If not, back to step 2 and resolve the equation with new iteration.

Value iteration algorithm is a method for determining the optimal strategy over finite and infinite – time horizon. At first iteration, we have h=0, all values of any state are initialized to 0. After that, the successive iteration are run by increasing one iteration. In each state (s) and for each action (x), the value of F(s,x) is calculated as the sum of all values that are associated with any next state in the future s' computed at the previous iteration (h-1). $F_{h-1}(s')$ Equals to $F_h(s, \pi_h(s))$. $\pi_h(s)$ Refers to the strategy that is selected within the iteration i. successive iterations are run forward until the resulting strategy converges toward an optimal solution. That is, when the difference between values from one iteration to the values of previous iteration for each states is equal or less than ε. And that refers to the optimal strategy π^*.

5. Computational Result:

The study was made at Al-Raji Group companies and it is one of the popular companies in Iraq for the production of soft drinks, juices, energy drinks and healthy water. Our focus was on production process and superintendence machines. The company had (6) number of machines and wants to allocate the machines to (4) number tasks, these machines are subject to fail and breakdown. The company had estimated that 0.95 of breakdown machines would join to the functional ones at each period. This paper aims to determine the minimum expected time for the completion of all the machines assigned to perform their tasks in the company by using the dynamic optimization technique during the H period of study. After calculating the probability of failure for all tasks from equation (6) and equation (7) which is denote to R(s) that is after recording the running times until failure occurs we obtain the results in the table below (1).

Table (1). Represents the probability that each task will fail

R(i) \ Task	1	2	3	4
$R(t_0)$	1	1	1	1
$R(t_1)$	0.8571	0.8333	0.8571	0.8
$R(t_2)$	0.7143	0.6667	0.7143	0.6
$R(t_3)$	0.5714	0.5	0.5714	0.4
$R(t_4)$	0.4286	0.3333	0.4286	0.2
$R(t_5)$	0.2857	0.1667	0.2857	
$R(t_6)$	0.1429		0.1429	
R(S)	0.0061	0.0154	0.0061	0.0384

And Table (2) gives detailed information of the expected initial completion times (in hours) for the machines to complete their tasks assigned to with the probability of failure for each task. The goal is to determine the minimum expected time for the completion of all the machines assigned to perform their tasks for each product in the company using the dynamic optimization method.

Table (2). The expected initial completion times of the machines in hours and the probability of failure machines

Machines \ Tasks	x^1	x^2	x^3	x^4
1	1.0667	1.1429	1.2	1.3714
2	1.0667	1.1429	1.2	1.3714
3	1	1	1	1
4	0.8772	0.8772	0.9524	1.1905
5	1.0033	1.0033	1.0033	1.0033
6	1.2	1.2973	1.2	1.3714
Probability of failure (R_k)	0.0061	0.0154	0.0061	0.0384

Assuming s_h represents machines number to be allocated at the next period, that means $s_h = x_h^k(s_{h-1}(b))$. Let s_0 be the number of machines at the beginning of the planning horizon. The number breakdown machines for all tasks ($\sum_{k=1}^{m} R_k x_h^k$). And expected to join the functional machines in the next period of process is ($0.05 * \sum_{k=1}^{m} R_k x_h^k$). We have the state of machines that,

$$s_h = s_{h-1} - \left((0.05) * \sum_{k=1}^{4} R_k x_h^k \quad , h = 1,2,\dots,H \right) \qquad (12)$$

Which is our transformation equation and is a stochastic variable. We can express our expected return function as follows:

$$Y_h^{\pi}(s_h) = E\left[\min_{x_h \in X(s_h)} \sum_{h=0}^{H} \sum_{k=1}^{4} \beta^h \Omega_h^k \left(x_h^{\pi^k}(s_{h-1}(b)) \right) \right] \qquad (13)$$

Subject to

$$\sum_{k=1}^{4} x_h^{\pi^k}(s_{h-1}(b)) = s_{h-1}, \quad h = 0,1,2,\dots,H$$

The company cannot allocate negative resource to any one task, so we write $x_h^{\pi^k}(s_{h-1}(b)) \geq 0, h = 0,1,\dots,H; k = 1,2,3,4$. Of course equation (13) is the same as

$$F_H(s_{h-1}) = \min_{x_h \in X(s_h)} \left[\sum_{k=1}^{4} \Omega_h^k \left(x_h^k(s_{h-1}(b)) \right) + E\{F_{h-1}(s_h)\} \right] \Rightarrow$$

$$F_H(s_{h-1}) = \min_{x_h \in X(s_{h-1})} \left[\sum_{k=1}^{4} \Omega_h^k \left(x_h^k(s_{h-1}(b)) \right) + E(F_{H-1}(s_{h-1} - (0.05 \sum_{k=1}^{4} R_k x_h^k))) \right]$$

Subject to

$$\sum_{k=1}^{4} x_h^k(s_{h-1}(b)) = s_{h-1},$$

$$x_h^{\pi^k}(s_{h-1}(b)) \geq 0, h = 0,1,\dots,H, k = 1,2,3,4$$

We have 4 tasks and 6 machines in our problem. Hence, m = 4. n = 6, by setting ε = 0.001.

Which is parametric linear programming problem with 4 variables. To solve (13), we used MATLAB program, the results obtained. At the end, we obtained the following results, which are shown in Table (3) below:

Table (3). Represents the results obtained and for several iterations using the value iteration algorithm which are solved by using MATLAB program.

		$F_h^*(s_h)$			
n.o. iter.	n.o. mach.	Task 1	Task 2	Task 3	Task 4
41	1	43.7347	43.8109	43.8680	44.0394
	2	43.7347	43.8109	43.8680	44.0394
	3	41	41	41	41
	4	35.9652	35.9652	36.0404	36.2785
	5	41.1353	41.1353	41.1353	41.1353
	6	49.2	49.2973	49.2	49.3714
		$F_h^*(s_h)$			
n.o. iter.	n.o. mach.	Task 1	Task 2	Task 3	Task 4
135	1	144.0045	144.0807	144.1378	144.3092
	2	144.0045	144.0807	144.1378	144.3092
	3	135	135	135	135
	4	118.4220	118.4220	118.4972	118.7353
	5	135.4455	135.4455	135.4455	135.4455
		$F_h^*(s_h)$			
n.o. iter.	n.o. mach.	Task 1	Task 2	Task 3	Task 4
300	1	320.0100	320.0862	320.1433	320.3147
	2	320.0100	320.0862	320.1433	320.3147
	3	300	300	300	300
		$F_h^*(s_h)$			
n.o. iter.	n.o. mach.	Task 1	Task 2	Task 3	Task 4
1150	1	1226.7	1226.8	1226.8	1227
		$F_h^*(s_h)$			
n.o. iter.	n.o. mach.	Task 1	Task 2	Task 3	Task 4
∞	1	1379.2	1379.3	1379.4	1379.5

In order to solve the problem and obtain the optimal policy and through the successive iteration. At the beginning of the study, all the values of the function and for all states are considered the initial value function F_0 *(s) = 0*. And after solving the problem using the forward method by repeating the successive iterative operations, the value function *F(s)* is an incremental function and as we proceed forward by repeating the process of iterations, the initial values will be accumulated at each stage. We note from the results that we obtained which are mentioned in the table (3) that in the 41st iteration, the first and second machines for the first task took about 43.7 hours, while for the second task they took about 43.8 hours. As for the third task, they took about 43.9 hours, and finally for the fourth task they took about 44 hours. The third and fifth machines had equal completion times for all tasks, for the third machine it took 41 hours and for all tasks, and for the fifth machine it took approximately 41.1 hours for all tasks. The completion time for the fourth machine and for the first, second and third tasks was approximately 36 hours, while they took about 36.3 hours for the fourth task. Finally, for the sixth machine, the time for completing the first and third tasks was equally same completion; it took approximately 49 hours, and it took 49.3 hours for the second task. As for the fourth task, it took about 49.4 hours. Figure (1) shows the results that we obtained at the 41 iteration.

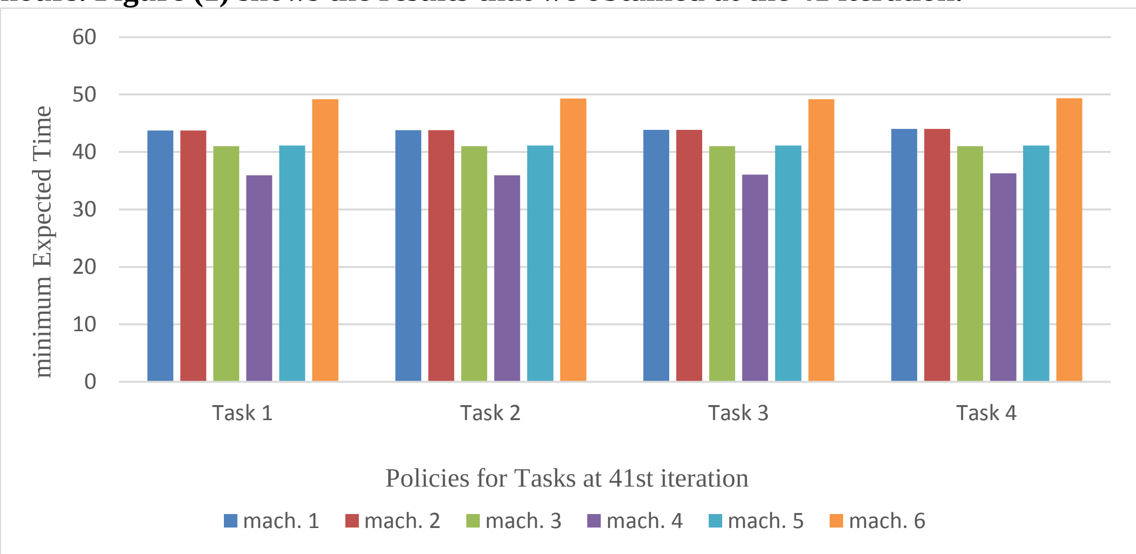

Fig (1). The expected time and the optimal policies that accrued at 41st iteration

After moving forward and by repeating the iterative operations, in the iteration 135 from Table (3), we notice that the number of machines became 5 machines, and in the iteration 300 the number of machines became 3because, according to the equation (12), s_h is decreasing. By the failure rate $\beta = 0.05$ and the probability of failure for each task *R(s)*. As for the first task of 300 repetitions, the first and second machines took about 320 hours, and the third machine took 300 hours. For the second and third task, the first and second machines took about 320.1 hours, and the third machine took 300 hours, while the first and second machines took about 320.3 hours for the fourth task, and the third machine took about 300 hours. Also Figure (2) below shows the results at iteration 300.

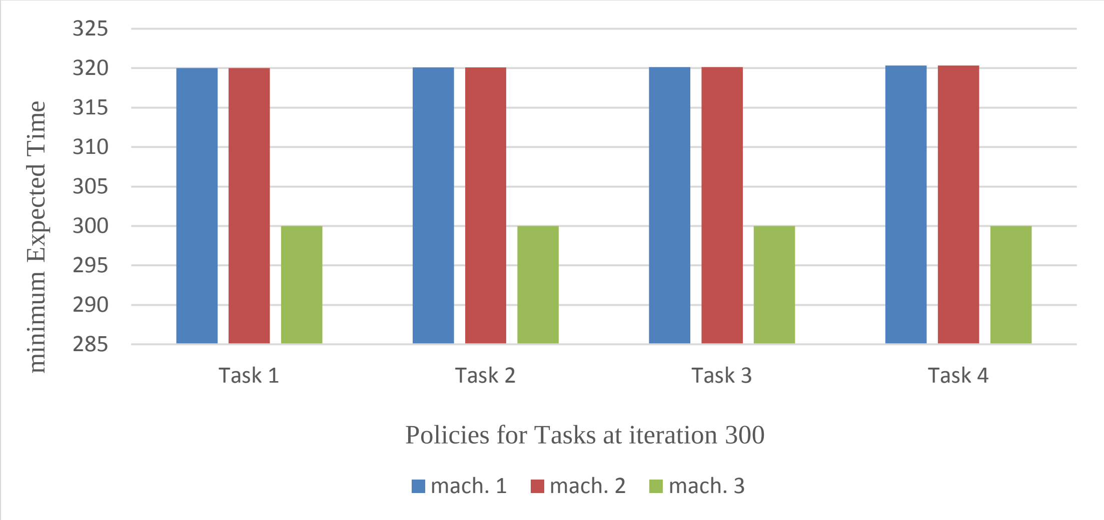

Fig (2). The expected time and optimal policies that accrued at iteration 300.

In the 1150 repetition, we notice that the first machine was taken for the first task, as it took about 1226.7 hours, and about 1226.8 hours for the second and third tasks, and about 1227 hours for the fourth task, as is shown in Figure (3)

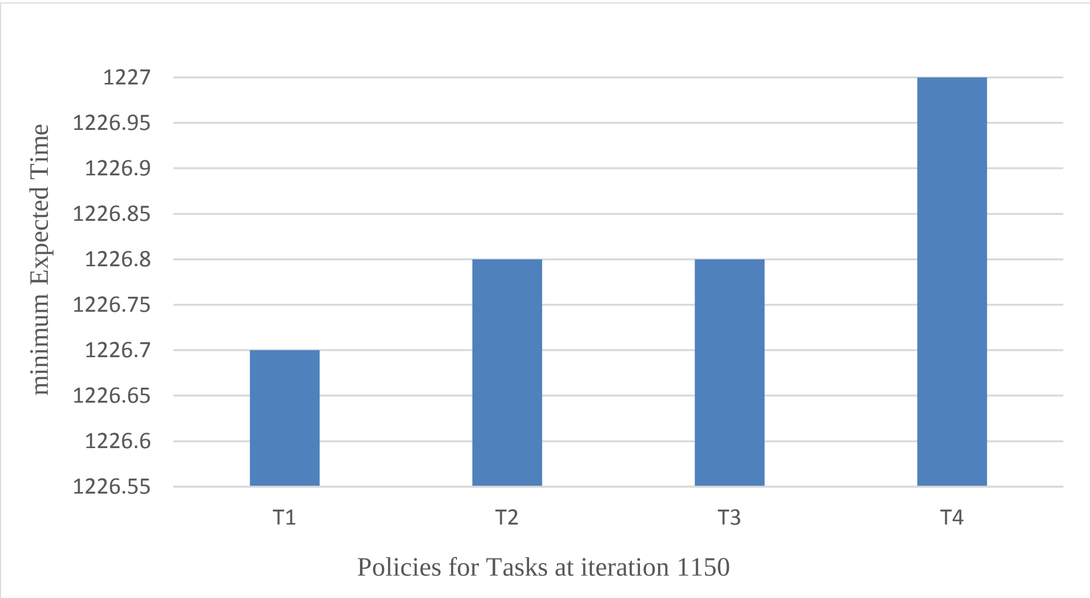

Fig (3). The expected time and optimal policies that accrued at iteration 1150.

And Figure (4) shows the results at the infinite of repetitions; the optimal solution has been stopped. The first and second tasks were equal in completing their tasks and took about 1379.2 hours and 1379.3 hours respectively, while the third task took about 1379.4 hours, and the fourth task took about 1379.5 hours.

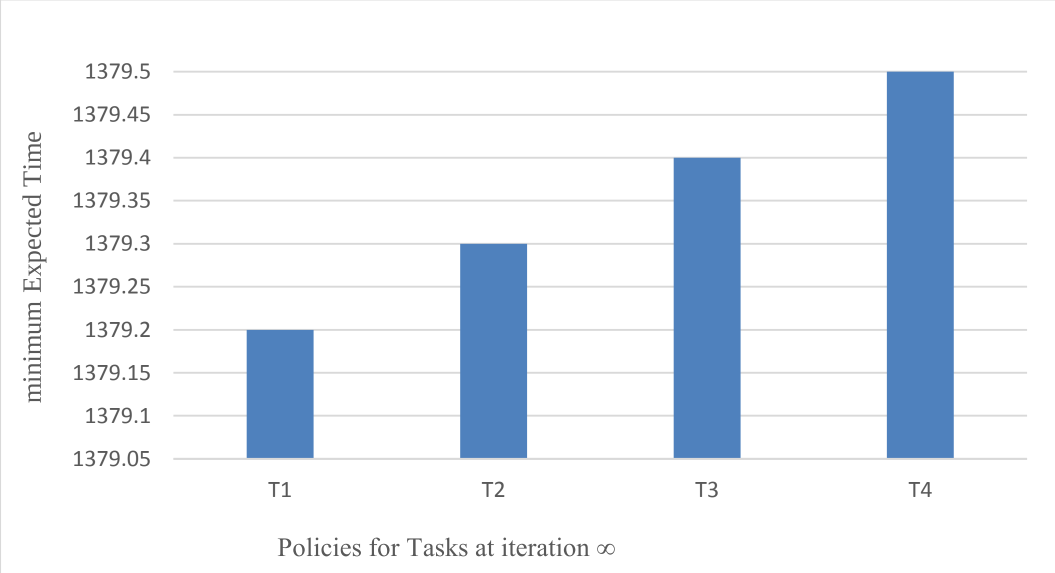

Fig (4). The expected time and optimal policies that accrued at iteration ∞.

6. Conclusion:

Many production companies have been allocating resources to different tasks without taking into account some of the factors that may hinder the achievement of their goals. This paper dealt with allocating machines to different tasks and comparing them with each other in order to reduce the time of completion of the machines for the tasks assigned to them and over a finite and infinite horizon. Through these results, we found in the previous section, we noticed that the period of completion of the machines became close to each other after it stopped at $s_\infty = 0$, because the company could not allocate negative machines. The first task took a few minutes shorter than the other tasks, and this means that the best policy that can be followed is for the first task and the second one. A careful analysis of the situation reveals that adequate planning and quick and effective maintenance can enhance the capability of the machines and thus reduce the time and effort, which contribute to reducing the company's costs and thus maximizing the production capability to increase the company's profits.

References:

1. Ajofoyinbo, A. M., & Orolu, K. O. (2012), "Optimal Allocation of Radio Resource in Cellular LTE Downlink Based on Truncated Dynamic Programming under Uncertainty", Int. J. Communications, Network and System Sciences, 5, p.p. 111-120.
2. AL-Nasser, A. H. (2009), "Reliability of System", ITHRAA Publishing and distribution, An introduction to Statistical Reliability, Amman, Jordan, p.p. 49-67.
3. Amuji, H. O., Ugwuanyim, G. U., Ogbonna, C. J., Iwu, H. C., & Okechukwu, B. N. (2017), "The Usefulness of Dynamic Programming in Course Allocation in the Nigerian Universities", Open Journal of Optimization, 6(4), p.p. 176-186.
4. D. P. de Farias & B. Van Roy. (2003), "The Linear Programming Approach to Approximate Dynamic Programming", Operations Research, 51(6), p.p. 850-865.
5. Ebeling, C. E. (2003)," Data Collection and Empirical Methods", University of Dayton, McGraw-Hill, An Introduction to Reliability and Maintainability Engineering, p.p. 283-307.
6. Ebeling, C. E. (2003)," Reliability of Systems", University of Dayton, McGraw-Hill, An Introduction to Reliability and Maintainability Engineering, p.p. 83-107.
7. G. L. Nemhauser & Z. Ullmann. (1969), "Discrete Dynamic Programming and Capital Allocation", Management Science, 15(9), p.p. 494-505.
8. Godfrey, G. A., & Powell, W. B. (2002), "An adaptive dynamic programming algorithm for dynamic fleet management, I: Single period travel times", Transportation Science, 36(1), p.p. 21-39.
9. Guestrin, C., Koller, D., Parr, R., & Venkataraman, S. (2003), "Efficient solution algorithms for factored MDPs", Journal of Artificial Intelligence Research, 19, p.p. 399-468.
10. Hauskrecht, M., & Singliar, T. (2012), "Monte-Carlo optimizations for resource allocation problems in stochastic network systems", In Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence, p.p. 305-312.
11. Hillier, F. S. (2001), "Dynamic Programming", Seventh Edition, McGraw-Hill, Introduction to Operations Research, p.p. 533-575.
12. Hoang, K. D., Hou, P., Fioretto, F., Yeoh, W., Zivan, R., & Yokoo, M. (2017), "Infinite-horizon proactive dynamic DCOPs", In Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, p.p. 212-220.
13. Mulvey, J. M., & Vladimirou, H. (1992), "Stochastic network programming for financial planning problems", Management Science, 38(11), p.p. 1642-1664.
14. Nkeki, C. I. (2013), "Dynamic Optimization Technique for Distribution of Goods with Stochastic Shortages. Journal of Optimization", Hindawi Publishing Corporation, Journal of Optimization, Volume 2013, p.p. 1-12.
15. Noor-E-Alam, M., & Doucette, J. (2012)," An application of infinite horizon stochastic dynamic programming in multi-stage project investment decision-making", International Journal of Operational Research, 13(4), 423-438.
16. Nwozo, C. R., & Nkeki, C. I. (2012), "On a dynamic optimization technique for resource allocation problems in a production company", American Journal of Operations Research, 2, p.p. 357-363.

17. Osman, M. S., Abo-Sinna, M. A., & Mousa, A. A. (2005),” An effective genetic algorithm approach to multiobjective resource allocation problems (MORAPs)”, Applied Mathematics and Computation, 163(2), 755-768.
18. Paul J. Schweitzer & Abraham Seidmann. (1985), “Generalized Polynomial Approximations in Markovian Decision Processes”, Journal of Mathematical Analysis and Applications, p.p. 568-582.
19. Powell, W. B. (2011), Approximate Dynamic Programming: Solving the curses of dimensionality, Princeton University, John Wiley & Sons, Vol. 842.
20. Powell, W. B. (2004), Approximate Dynamic Programming for Asset Management, Princeton University, Princeton.
21. Powell, W. B., & Topaloglu, H. (2003), “Stochastic programming in transportation and logistics”, Handbooks in operations research and management science, Elsevier, Amsterdam, p.p. 555-635.
22. Powell, W. B., George, A., Bouzaiene-Ayari, B., & Simao, H. P. (2005), “Approximate dynamic programming for high dimensional resource allocation problems”, In Proceedings In Proceedings, 2005 IEEE International Joint Conference on Neural Networks, Vol. 5, p.p. 2989-2994.
23. Puterman, M. L. (1990), Markov Decision Processes Discrete Stochastic Dynamic Programming, John Wiley & Sons, Hoboken, New Jersey.
24. Spivey, M. Z., & Powell, W. B. (2004), “The dynamic assignment problem”, Transportation Science, 38(4), p.p. 399-419..
25. Topaloglu, H., & Kunnumkal, S. (2006), “Approximate dynamic programming methods for an inventory allocation problem under uncertainty”, Naval Research Logistics (NRL), 53(8), 822-841.
26. Topaloglu, H., & Powell, W. B. (2006), “Dynamic-programming approximations for stochastic time-staged integer multi-commodity flow problems”, Informs Journal on Computing, 18(1), p.p. 31-42.
27. Van Roy, B., Bertsekas, D. P., Lee, Y., & Tsitsiklis, J. N. (1997), “A neuro-dynamic programming approach to retailer inventory management”, In Proceedings of the 36th IEEE Conference on Decision and Control, Vol. 4, p.p. 4052-4057.
28. Venner, S., Chadès, I., Bel-Venner, M. C., Pasquet, A., Charpillet, F., & Leborgne, R., (2006),” Dynamic optimization over infinite-time horizon: web-building strategy in an orb-weaving spider as a case study”, Journal of theoretical biology, 241(4), P.P. 725-733.
29. Zhou, J., Jiang, Z., Qin, H., Ji, C., & Feng, Z., (2017), “Two dimension reduction methods for multi-dimensional dynamic programming and its application in cascade reservoirs operation optimization”, Water, 9(9), 634, P.P. 1-19.

حل نموذج تخصيص الموارد بأستعمال أسلوب الأمثلية الديناميكية لمجموعة شركات الراجي للمشروبات الغازية و العصائر

الباحث/ محمد حسن
كلية الأدارة و الأقتصاد / جامعة بغداد / العراق
Mohammed.hasan1101a@coadec.uobaghdad.edu.iq

أ.د. محمد صادق عبدالرزاق الدوري
كلية الأدارة و الأقتصاد / جامعة بغداد / العراق
dr_aldouri@coadec.uobaghdad.edu.iq

Received:22/3/2021 Accepted: 10/5/2021 Published: SEPTEMBER/ 2021

المستخلص:

في هذا البحث, تمت دراسة مشكلة تخصيص الموارد في شركة الراجي للمشروبات الغازية والعصائر. الشركة لديها عدة مهام تحتاج لإنجازها لتنتج عدة أنواع من العصائر و المشروبات الغازية و التي تحتاج الى مكائن لتنجز هذه المهام، إذ لديها عدد 6 مكائن ترغب بتخصيصها على 4 مهام مختلفة لإنجاز هذه المهام. المكائن المخصصة لكل مهمة تكون معرضة للفشل، إذ يتم تصليح هذه المكائن لتشترك مرة أخرى في العملية الإنتاجية. من السجلات السابقة للشركة ، تم حساب احتمال فشل المكائن في كل مهمة بالإعتماد على بيانات معلومات الشركة. و كذلك تم تسجيل الوقت الذي تتطلبه كل ماكنة لانجاز كل مهمة. الهدف من هذا البحث هو تحديد الحد الأدنى من الوقت المتوقع لإكمال جميع المكائن المخصصة لأنجاز مهامهم في الشركة باستخدام أسلوب الامثلية الديناميكي عبر آفاق محدودة وغير محدودة. من خلال مقارنة النتائج فقد تبين أن المهمتين الأولى والثانية كانتا أفضل من المهمتين الثالثة والرابعة لأن المهمة الأولى والثانية أكملت مهامهما في فترة أقصر من المهمتين الثالثة و الرابعة، فقد استغرقتا 1379.2 ساعة و 1379.3 ساعة على التوالي تقريبا خلال $H=\infty$ من الآفاق (المراحل)، في حين أن المهمة الثالثة اخذت وقت 1379.4 ساعة و المهمة الرابعة 1379.5 ساعة. كشف التحليل الدقيق للوضع أنه تم تخفيض المدة التي تستغرقها كل ماكنة لإنجاز مهامها و أن التخطيط المناسب والصيانة السريعة والفعالة يمكن أن تعزز قدرة المكائن وبالتالي تقلل الوقت والجهد، مما يساهم في تقليل تكاليف الشركة وبالتالي تعظيم القدرة الإنتاجية لزيادة أرباح الشركة.

المصطلحات الرئيسة للبحث/ **مشكلة تخصيص الموارد، الأمثلية الديناميكية، الافق النهائي و اللانهائي، خوارزمية تكرار القيمة.**

*البحث مستل من رسالة ماجستير