Research Annals of Industrial and Systems Engineering

www.raise.reapress.com

Res. Ann. Ind. Syst. Eng. Vol. 1, No. 4 (2024) 236–243.

Paper Type: Original Article

Applying the DEA Model to Evaluate FPGA Devices Running AES Algorithm: A KNIME-Based Approach

Muhamad Ali Haji Soltani[1,*], Mehdi Navaei Leilan[2]

[1] Department of Law, Shahid Beheshti University, Tehran, Iran; malihaji1377@gmail.com.
[2] Tabriz University, Tabriz, Iran; m.navaei2024@gmail.com.

Citation:

Received: 02 September 2024 *Revised: 01 November 2024* *Accepted: 30 December 2024*	Haji Soltani, M. A., & Navaei Leilan, M. (2024). Applying the DEA model to evaluate FPGA devices running AES algorithm: A KNIME_based approach. *Research annals of industrial and systems engineering, 1*(4), 236-243.

Abstract

The effectiveness of Advanced Encryption Standard (AES) algorithm implementations on Field Programming Gate Array (FPGA) circuits is examined in this paper using Data Envelopment Analysis (DEA). DEA offers insights beyond conventional ratio analysis by examining various input-output scenarios, locating optimal practices, and maximizing resource allocation. We evaluate different implementations using the Charnes, Cooper, and Rhodes (CCR) model in the KNIME framework, emphasizing the significance of ongoing hardware security enhancement to address changing data protection concerns.

Keywords: Data envelopment analysis, Advanced encryption standard, Field programming gate array, KNIME.

1 | Introduction

As the use of optimal systems becomes more and more important, there is a growing need for a tool to check and analyze the optimality and efficiency of these systems. Data Envelopment Analysis (DEA) [1] is a technique that enables the measurement of the relative efficiency of organizational units known as Decision-Making Units (DMUs). The methodology's main strength lies in its ability to capture the interplay between multiple inputs and outputs, a process that cannot be satisfactorily probed through traditional ratio analysis. Knowing how efficiency is measured goes beyond simple calculations and ratios; it entails a careful analysis of the ways in which different inputs and outputs interact to show the actual effectiveness of DMUs. This intricacy is frequently not adequately captured by traditional ratio analysis, which leaves gaps in our evaluation of overall efficiency. In addition to improving decision-making, this all-inclusive framework helps organizations make better choices, and designers create more effective designs.

Corresponding Author: malihaji1377@gmail.com
https://doi.org/10.22105/raise.v1i4.64

The world's technical breakthroughs, the constantly growing number of developing technologies, and the development of global and self-determining systems have made the need for improved data security increasingly evident. The use of symmetric key encryption algorithms, particularly the Advanced Encryption Standard (AES) algorithm, is one of the primary ways to preserve secrecy, integrity, and authentication of data access [2]. These algorithms provide a robust framework for protecting sensitive information from unauthorized access while ensuring that legitimate users can securely encrypt and decrypt data. As cyber threats continue to evolve, organizations must adopt comprehensive cybersecurity strategies that incorporate advanced encryption techniques alongside other protective measures. Hardware implementation of the encryption techniques is required to do this. Field Programming Gate Array (FPGA) chips are among the greatest and most efficient platforms and tools for putting encryption methods into practice [3]. Because of their wide range of applications and high programming flexibility, FPGAs have emerged as a popular tool for this implementation. However, determining the efficacy of the method used on FPGA is one of the current difficulties. One of the most effective methods for determining if such systems are optimal is the DEA technique. This approach allows for the assessment of the relative efficiency of different encryption implementations on FPGAs by comparing various input-output scenarios. By applying DEA, researchers can identify best practices and optimize resource allocation, ultimately enhancing the performance and security of encryption methods in hardware.

The DEA technique was initially introduced by Charnes et al. [4], followed by another model for this calculation presented in [5]. In this article, we have employed this efficiency calculation technique on the model we previously proposed in [6]. Additionally, we have applied this model to [7], [8], and [9] for comparative analysis with other implementations. The method we utilized for this calculation is the Constant Returns to Scale (CRS) model outlined in [4], which evaluates the efficiency of our proposed model. This approach enables a systematic comparison of our results with those derived from alternative methodologies, thereby enhancing the robustness of our findings. By employing the CRS model, we can identify best practices among the evaluated units and pinpoint areas for potential improvement.

The DEA technique will be calculated on the models that are presented using Python and KNIME software, which is a powerful data analysis tool that can be used to execute a variety of workflows. KNIME Software's versatility makes it a valuable tool in the field of data processing, and its visual implementation makes it easy to construct complex workflows, making it accessible to both novice and experienced data scientists [10]. The platform's high flexibility allows for the seamless integration and combination of complex workflows, allowing users to customize their processes to meet particular analytical needs. As an open-source tool, KNIME not only offers advanced coding capabilities for those seeking deeper customization but also fosters a collaborative environment where users can share insights and innovations. Furthermore, the ability to create graphics and Block diagram representations enhance understanding and communication within teams, ultimately driving more effective decision-making. Together, these features underscore KNIME's powerful role in transforming data into actionable intelligence.

2 | Preliminaries

The DEA method is a productivity calculation technique that is used to determine the efficiency of DMUs. The advantage of this calculation method is that it can be used to determine the efficiency of any type of DMU, be it an economic model, an organization, or an FPGA chip. This adaptability enables a thorough analysis across a variety of sectors, facilitating knowledgeable decision-making and resource allocation. By comparing themselves to the most efficient units, organizations can pinpoint areas for improvement and put plans in place to increase overall productivity. After applying the DEA technique, a DMU unit is deemed technically efficient if it uses the least amount of inputs for the maximum outputs or, to put it another way, if it has the minimum level of input to achieve a specified level of output. The DEA technique is calculated on data by looking at DMU units, inputs, and outputs.

When we use the DEA technique to calculate the efficiency of each DMU in the dataset, the best and most efficient DMUs are referred to as the efficient frontier. One benefit of this technique is that it only requires existing data for calculations. The DEA model uses Linear Programming (LP) for efficiency evaluation, which entails creating a mathematical model for each DMU and comparing each DMU in terms of inputs and outputs by solving a system of linear inequalities. An inefficient DMU is one whose performance can be improved by lowering inputs without changing outputs or by increasing outputs without changing inputs.

The DEA technique can be implemented using one of two primary models. Based on DMU performance under CRS, the Charnes, Cooper, and Rhodes (CCR) model is the simplest model for implementing the DEA approach [4]. This implies that outputs will rise proportionately when a given factor increases inputs. The Banker, Charnes, Cooper (BCC) model, the alternative, makes the assumption that DMUs will function under Variable Returns to Scale (VRS), which means that changes in the performance factor will affect efficiency [5]. Since an increase in inputs does not necessarily translate into a corresponding rise in outputs, this model's strength is its greater resemblance to real-world applications.

When a linear and proportionate relationship between inputs and outputs is anticipated, the CCR model is especially helpful. This model is frequently used in settings where DMUs or organizations function at an ideal scale, making it possible to evaluate efficiency with ease. By comparing the input-output ratios of DMUs, we can use the CCR model to assess their relative performance and distinguish between those that function well and those that don't.

Therefore, taking into account the explanations given, we will compute the efficiency of the DMU units in the models shown in [6], [7], [8] and [9] using the CCR technique for simplicity. Based on the assumption that each FPGA is a DMU, the inputs and outputs will be identified and subjected to computations.

Eq. (1) may be used to describe the efficiency of the DMU_k based on the CCR model. This DMU is identical to *Fig. 1* in that it has m inputs $x_{1k}, x_{2k}, \dots, x_{mk}$ with linear coefficients $v_1, v_2, \dots, v_m$ and n outputs $y_{1k}, y_{2k}, \dots, y_{nk}$ with linear coefficients $u_1, u_2, \dots, u_m$.

$$\max_{u_i} r_k = \frac{\sum_i^n u_i y_{ik}}{\sum_j^m v_j x_{jk}},$$

$$\text{subject to: } r_l = \frac{\sum_i^n u_i y_{il}}{\sum_j^m v_j x_{jl}} \leq 1, \qquad \text{where: } l = 1,2,\dots,k, \tag{1}$$

$$u_i \,\&\, v_j \geq 0 \quad \text{for } i \,\&\, j = 1,2,\dots,n.$$

In which r_k is the efficiency of DMU_k, and the goal is to maximize u_i, in such a way that the value of r_k is maximized.

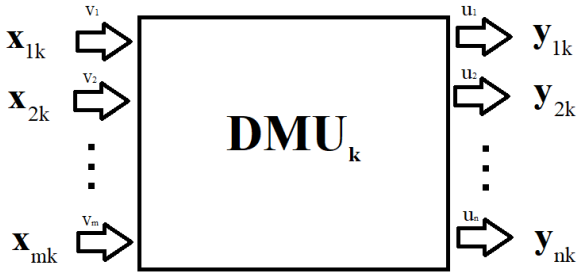

Fig. 1. Decision-making unit with multiple inputs and outputs.

A DMU can have several inputs and outputs, as shown in *Fig. 1*. The general layout of each FPGA unit as a DMU will be as illustrated in *Fig. 2*, taking into account the information listed in *Table 1* as the inputs and outputs of an FPGA unit.

Table 1. Implementations of advanced encryption standard on field programming gate array.

Design	Reg.	LUT.	BRAM	Rand.Bits	CLK_{MHZ}	Lat.	$Time_{\mu s}$	$Thr.put_{Gbit/s}$	Device
[6]	526	3728	120	160	105	50	0.476	1.344	Spartan-6
E [7]	527	2068	240	160	116	50	0.431	1.484	Spartan-6
E/D [7]	527	3778	240	160	102	50	0.490	1.305	Spartan-6
[8]	1566	2888	16	51.2	100	512	5.12	0.025	Virtex-II pro
[9]	5328	7783	0	0	131	70	0.534	1.676	Spartan-6

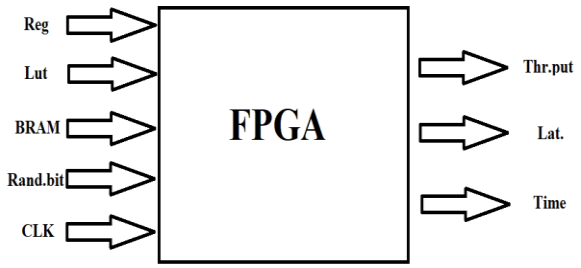

Fig. 2. Field programming gate array with inputs and outputs.

Although the five AES algorithm implementations on five FPGA chips in *Table 1* have different specifications, the resources used in each implementation are listed. Based on *Table 1*, each FPGA chip's input data is shown in *Fig. 2*. This data contains the Clock (CLK), Lookup Tables (LUTs), Block RAMs (BRAMs), random bits, and the number of registers. These details are crucial for understanding the performance and efficiency of each implementation.

I. FPGA registers are quick, compact storage components that are used to store data while it is being processed temporarily. Registers are essential for a number of activities, including storing intermediate computation results, controlling data flow between circuit components, and coordinating operations in sequential logic circuits.

II. LUTs in FPGAs are versatile components that can be configured to implement various logic functions by mapping input combinations to output values. In the context of encryption algorithms, LUTs play a crucial role by enabling the efficient implementation of complex mathematical operations and substitution processes, which are fundamental to many cryptographic techniques. They allow for rapid data transformation and can be used to create Substitution Boxes (S-boxes) that enhance security by obscuring the relationship between plaintext and ciphertext.

III. BRAMs in FPGAs are dedicated memory resources that provide high-speed storage for data and instructions, allowing for efficient data processing and management within digital designs. They are essential for applications requiring large amounts of memory, such as buffering, data caching, and implementing complex algorithms.

IV. Random bits are crucial for implementing encryption algorithms on FPGAs as they provide the necessary unpredictability and security for cryptographic processes. These bits are often used to generate keys, initialization vectors, and nonces, which are essential for ensuring that encrypted data remains secure against attacks.

V. The CLK in an FPGA chip serves as a timing reference that synchronizes the operation of all digital circuits, ensuring that data is processed and transferred in a coordinated manner across the device.

VI. The time lag between the plaintext input and the ciphertext output of an encryption implementation on an FPGA is known as the latency, and the design architecture, CLK frequency, and complexity of the

encryption algorithm impact it. "Time" refers to more general durations associated with the encryption process as a whole, whereas latency concentrates on the delay of a specific operation.

VII. The throughput of an implemented encryption algorithm on an FPGA device refers to the rate at which data can be processed and encrypted, typically measured in Bits Per Second (bps). Optimizing the design and resource utilization of the FPGA can lead to improved throughput, making it suitable for high-performance cryptographic applications.

Register, LUT, BRAM, random bits, and CLK are the inputs, as can be seen, and throughput, time, and latency are the outputs of these implementations.

3|Applying the Data Envelopment Analysis Technique

When computing the DEA technique on these models, it is crucial to keep in mind that, when looking at the outputs of these DMUs, it is preferable to increase throughput while decreasing latency and time for the best possible implementation. As a result, we have chosen to use the inverse factors of latency and time for computations in our technique. In other words, *Table 1* will be rebuilt as *Table 2* since we view the components 1/latency and 1/time as outputs.

To put it another way, we took time and latency values into account as inverse factors among the output variables because the technique's objective in the context of CCR is to maximize outputs while maintaining constant inputs. Two of the three distinctive output components of the FPGA version of the AES algorithm are time and latency; for the best results, it is preferable to decrease rather than increase these variables' values. In order to calculate and use the DEA technique as a straightforward solution, we have used the inverse of these output components as a relative numerical factor in this study.

So, this approach not only enhances the clarity of our analysis but also aligns with the overarching goal of optimizing performance metrics within the DEA framework. The results are shown in *Table 2*.

Table 2. Implementations of advanced encryption standard on field programming gate array with reversed factors Lat. and time.

Design	Reg.	LUT.	BRAM	Rand.Bits	CLK_{MHZ}	1/Lat.	$1/Time_{\mu s}$	$Thr.put_{Gbit/s}$	Device
[6]	526	3728	120	160	105	0.02	2.1008	1.344	Spartan-6
E [7]	527	2068	240	160	116	0.02	2.3201	1.484	Spartan-6
E/D [7]	527	3778	240	160	102	0.02	2.0408	1.305	Spartan-6
[8]	1566	2888	16	51.2	100	0.0019	0.1953	0.025	Virtex-II pro
[9]	5328	7783	0	0	131	0.0142	1.8726	1.676	Spartan-6

We used the KNIME software to compute the DEA approach on the current AES implementations on FPGA. KNIME is an open-source data analysis program. The ability to combine Python code with the many nodes and modules that KNIME offers is one benefit of utilizing it for analysis and calculation. This enables the development of an effective and visually appealing workflow.

The availability of comprehensive libraries for a variety of data analysis and optimization tasks, which give users quick and visible access to a large selection of data processing and machine learning techniques, is another benefit of utilizing KNIME. The development of independent and modular nodes for calculating the DEA process throughout the entire workflow is also made possible by the ease with which new nodes can be created using various programming languages and then combined. This is done by utilizing Python programming and the software's built-in tools.

We must now create a LP model to determine the efficiency of each DMU using the data in *Table 2* and *Eq. (1)*. The definition of the "Linprog" function in the Python SciPy library will be used to calculate this LP model, and *Eq. (2)* will be used to prepare the data for the LP model.

$$\min_x \; f^T x \;\text{ such that} \begin{cases} A \cdot x \le b, \\ A_{eq} \cdot x = b_{eq}, \\ l_b \le x \le u_b, \end{cases} \tag{2}$$

A and b relate to the definition of inequalities, A_{eq} and b_{eq} are the equality constraints, and l_b and u_b are the lower and upper bounds of the decision variable values. In *Eq. (2)*, f and x are the coefficients associated with the objective function and the decision variable, respectively. Thus, we will recast our model in the form of *Eq. (3)* based on *Eq. (2)*:

$$\begin{aligned} &\max u_i \sum_i^n u_i y_{ik}, \\ &\text{subjecct to: } \sum_j^m v_j x_{jk} = 1, \\ &\sum_i^n u_i y_{il} \le \sum_j^m v_j x_{jl}\ , \qquad l = 1,2,\dots,k, \\ &u_i \ge 0, \quad \text{for all i}, v_j \ge 0 \text{ for all j}. \end{aligned} \tag{3}$$

Since the linprog function's computing approach involves minimizing the objective function's coefficients, we have utilized the negative of the coefficients, or $(-1) * (\text{coefficients})$, in our calculations to carry out our calculations using this function. Thus, we create the vector f as indicated in *Eq. (4)* using the information in *Table 2* as well as *Eqs. (2)* and *(3)*.

$$f \text{ is a } (n+m) * 1 \text{ vector}, \qquad \text{where: } f_i = -y_{ik} \quad , i = 1,2,\dots,n. \tag{4}$$

Eq. (5) will be used to build matrix A:

$$A = \begin{bmatrix} y_{11} & y_{21} & \cdots & y_{n1} & -x_{11} & -x_{21} & \cdots & -x_{m1} \\ y_{12} & y_{22} & & y_{n2} & -x_{12} & -x_{22} & & -x_{m2} \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & \vdots \\ y_{1k} & y_{2k} & \cdots & y_{nk} & -x_{1k} & -x_{2k} & \cdots & -x_{mk} \end{bmatrix}. \tag{5}$$

Eq. (6) contains the values of A_{eq} and b_{eq}:

$$\begin{aligned} &A_{eq} \text{ is a } 1 * (n+m) \text{ vector}, \qquad \text{where: } A_{eq_{n+j}} = x_{jk} \quad , j = 1,2,\dots,m, \\ &b_{eq} = 1. \end{aligned} \tag{6}$$

The value of vector b is represented by *Eq. (7)*:

$$b \text{ is a } K * 1 \text{ vector}, \qquad \text{where: } b_i = 0 \quad , i = 1,2,\dots,K \quad \& \text{ K is the amount of DMUs.} \tag{7}$$

Lastly, *Eq. (8)* will be used to write the value of l_b and u_b:

$$l_b = 0 \quad \& \quad \text{no upper bound limit.} \tag{8}$$

The Python code for determining the linprog function for each DMU has been constructed with the aforementioned items in mind, and it will be included as a node to the entire workflow in KNIME, as shown in *Fig. 3*.

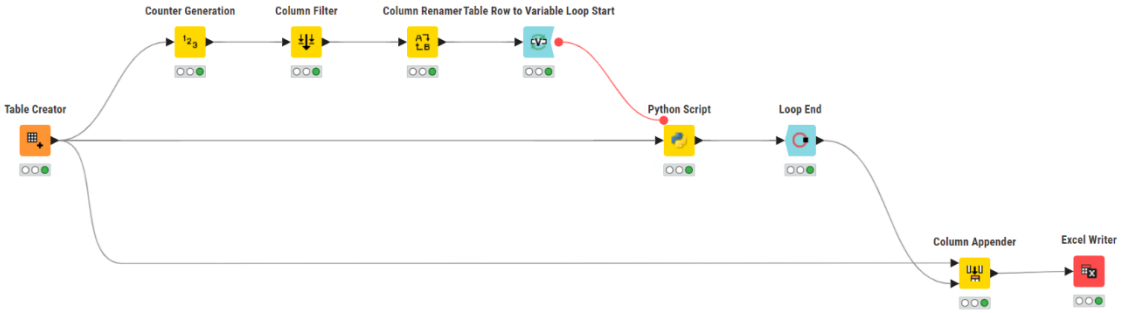

Fig. 3. KNIME workflow to calculate efficiency.

Table 3 displays the outcomes of this workflow's execution with reference to each DMU's efficiency. By using this method, it is observed that [6], [7], and [9] are effective and ideal implementations that make up the dataset's efficient frontiers.

Table 3. Implementations of advanced encryption standard on field programming gate array with efficiency.

Design	Reg.	LUT.	BRAM	Rand.Bits	CLK_{MHZ}	1/Lat.	$1/Time_{\mu s}$	$Thr.put_{Gbit/s}$	Device	Efficiency
[6]	526	3728	120	160	105	0.02	2.1008	1.344	Spartan-6	~1
E [7]	527	2068	240	160	116	0.02	2.3201	1.484	Spartan-6	~1
E/D [7]	527	3778	240	160	102	0.02	2.0408	1.305	Spartan-6	~1
[8]	1566	2888	16	51.2	100	0.0019	0.1953	0.025	Virtex-II pro	0.292495
[9]	5328	7783	0	0	131	0.0142	1.8726	1.676	Spartan-6	~1

The optimal solutions for these implementations are presented in *Table 4*.

Table 4. Optimal solution for implemented models.

Design	[6]	E [7]	E/D [7]	[8]	[9]
coefficients	0	50.0000000	50.0000000	149.75747300	70.0000014
	0	0	0	0	0
	0.74404762	0	0	0	0
	0	0	0.0001322	0.00040154	0.0001877
	0.00016022	0.0001856	0.0000004	0	0
	0.00335571	0.0025674	0	0.0231995	0.0108440
	0	0	0.0057722	0	0
	0	0	0.0000510	0	0
Efficiency	~1	~1	~1	0.292495	~1

4 | Conclusion

The efficiency of various hardware implementations of the AES algorithm on FPGA chips was investigated in this article using DEA techniques and the CCR model within the KNIME framework. DEA is a very useful technique for determining efficient systems and can be used as a tool for optimal decision-making, enabling the calculation of efficiency regardless of the type of DMUs. In the calculations conducted in this article, we reversed two of the three output factors to facilitate an increase in outputs while keeping inputs constant. However, the primary objective of the existing models for AES algorithm implementations on FPGA processors was to achieve a reduction in both latency and time. We classified implementations [6], [7], and [9] as belonging to the efficient frontier by carrying out these computations by creating a system of linear equations and inequalities and solving them. This allowed us to determine the best solutions and efficiency for each implementation.

The results also highlight the significance of ongoing assessment and improvement in the quickly developing field of hardware security. We can make sure that future implementations not only satisfy present performance criteria but also adjust to new problems in the field of secure data processing by utilizing sophisticated analytical tools like DEA.

Funding

This research was carried out without any financial support from public, commercial, or non-profit funding agencies.

Data Availability

All data supporting the findings of this study are included within the article. Additional information may be provided upon reasonable request.

Conflicts of Interest

No conflicts of interest have been declared in relation to this publication.

References

[1] Bowlin, W. F. (1998). Measuring performance: An introduction to data envelopment analysis (DEA). *The journal of cost analysis*, *15*(2), 3–27. https://doi.org/10.1080/08823871.1998.10462318

[2] Yassein, M. B., Aljawarneh, S., Qawasmeh, E., Mardini, W., & Khamayseh, Y. (2017). Comprehensive study of symmetric key and asymmetric key encryption algorithms. *2017 international conference on engineering and technology (ICET)* (pp. 1–7). IEEE. https://doi.org/10.1109/ICEngTechnol.2017.8308215

[3] Ruiz-Rosero, J., Ramirez-Gonzalez, G., & Khanna, R. (2019). Field programmable gate array applications—A scientometric review. *Computation*, *7*(4), 63. https://doi.org/10.3390/computation7040063

[4] Charnes, A., Cooper, W. W., & Rhodes, E. (1978). Measuring the efficiency of decision making units. *European journal of operational research*, *2*(6), 429–444. https://doi.org/10.1016/0377-2217(78)90138-8

[5] Banker, R. D., Charnes, A., & Cooper, W. W. (1984). Some models for estimating technical and scale inefficiencies in data envelopment analysis. *Management science*, *30*(9), 1078–1092. https://doi.org/10.1287/mnsc.30.9.1078

[6] Hajisoltani, M., Salarifard, R., & Soleimany, H. (2022). Secure and low-area implementation of the AES using FPGA. *Journal of information security*, *14*(3), 93–99. https://doi.org/10.22042/isecure.2022.14.3.0

[7] Shahmirzadi, A. R., Božilov, D., & Moradi, A. (2021). New first-order secure AES performance records. *Cryptology eprint archive*, *2021*(2), 304–327. https://doi.org/10.46586/tches.v2021.i2.304-327

[8] Güneysu, T., & Moradi, A. (2011). Generic side-channel countermeasures for reconfigurable devices. *Cryptographic hardware and embedded systems-ches 2011* (pp. 33–48). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-23951-9_3

[9] Shahmirzadi, A. R., & Moradi, A. (2021). Re-consolidating first-order masking schemes: Nullifying fresh randomness. *IACR transactions on cryptographic hardware and embedded systems*, *2021*(1), 305–342. https://doi.org/10.46586/tches.v2021.i1.305-342

[10] Delen, D. (2024). Landscape of tools for business analytics and data science--a tutorial on knime. *Proceedings of the 2024 pre-icis sigdsa symposium*. AIS Electronic Library (AISeL). https://aisel.aisnet.org/sigdsa2024/21