International Journal of Pure and Applied Mathematics
Volume 119 No. 12 2018, 13437-13452
ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

ijpam.eu

Survey on Hardware Implementation of Montgomery Modular exponentiation

K. Pratibha, Muthaiah Rajappa*

School of Computing, SASTRA Deemed University, India

***Corresponding Author**

Abstract

This paper gives the information regarding different methodology for modular multiplication with the modification of Montgomery algorithm. Montgomery multiplier proved to be more efficient multiplier which replaces division by the modulus with series of shifting by a number and an adder block. For larger number of bits, Modular multiplication takes more time to compute and also takes more area of the chip. Different methods ensure more speed and less chip size of the system. The speed of the multiplier is decided by the multiplier. Here three modified Montgomery algorithm discussed with their output compared with each other. The three methods are Iterative architecture, Montgomery multiplier for faster Cryptography and Vedic multipliers used in Montgomery algorithm for multiplication.Here three boards have been used for the analysis and they are Altera DE2-70, FPGA board Virtex 6 and Kintex 7.

Keywords: Montgomery algorithm, Modular multiplication (MM), Montgomery Modular Multiplication (MMM), Cryptography, Cryptography, cryptosystem, Urdhawa Tiryagbhayam Sutra and Montgomery Core.

1. Introduction

It requires advanced knowledge of theoretical computer science, number theory, algebra and combinational mathematics for evaluation and design of cryptosystem. There is always a need for the high speed implementation of the cryptosystem. Such example of cryptosystem is the Diffie-Hellman, RSA public key exchange cryptography. There are several issues which should be kept in mind while designing cryptosystem such as speed, security, key length and implementing board. For key exchange operation there is a need of modular operation which involves addition, multiplication and exponentiation and takes more time and implementation area. So for deducting

such issues, Montgomery modular multiplication has been used for large integer numbers. Now a day, in public key cryptography the key length is usually of the size of 1024 to 2048 bits. The poor timing characteristics can be utilized by the attackers to discover the entire private secret key. This issue forces developers to implement high-speed and space efficient algorithms.

In cryptosystem like Diffie-Hellman and RSA, calculating A.B mod p is very important operation and the computation of such operation takes more clock cycle as well as more chip area. Therefore, we go for reduction operation such as MMM [1].

***Algorithm**:Radix-2 MMM*

Inputs: P, A, B (n bits each), where B and A are less than P

r = 2; both M and r are prime to each other

$R = r^n$;

$P' = -P^{-1} \bmod r$.

Output: $C = A.B.R^{-1} \bmod P$

C=0;

For (i=0; i<n; i++)

$T = (C_0 + A_0.B_i).P' \bmod r$

$C = (C + A.B_i + P.T)/r$

End

If C>P

Then C=C-P

End

Return C

In this paper, the MMM algorithm has been discussed in different stages. Section 1 explains [2] Iterative architecture, section 2 explains [3] Montgomery multiplier for faster Cryptography and section 3 explains [4] Vedic multipliers used in Montgomery algorithm for multiplication.

2 .Iterative architecture

There are input-output limitations on DE2-70 board. It have only 475 I/O pin but for 2048-bits data path ,the modulo multiplications are designed. For overcoming such problem in designing of system, SIPO and PISO has been realized [2]. The 64-bits input are partitioned to the top level of the design and sent in serial with LSB first. The combinations of the partition which are 64-bits wide are the output of SIPO modules. The module containing PISO perform the opposite to the module containing SIPO.

The figures 1, 2 and 3 shows the Montgomery Multiplier core which consists of adders and multiplexers. Inside the Montgomery Core, the input results are processed in repetitive way.

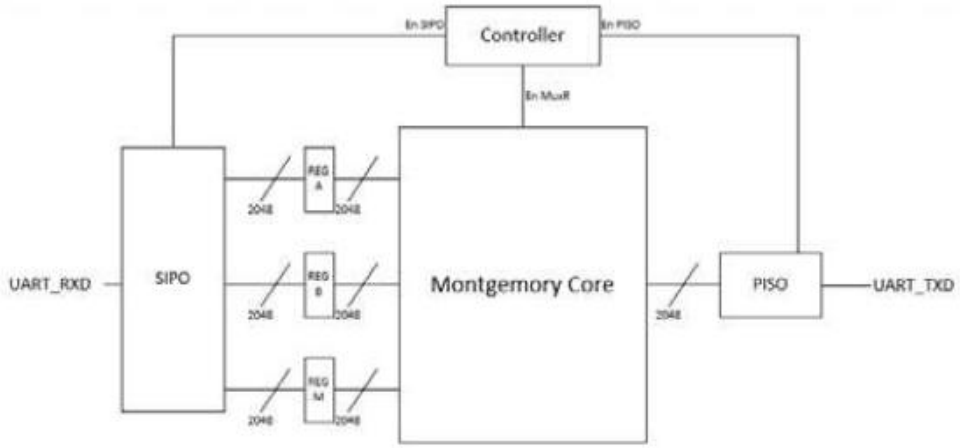

Figure -1 Block diagram(Top Level)

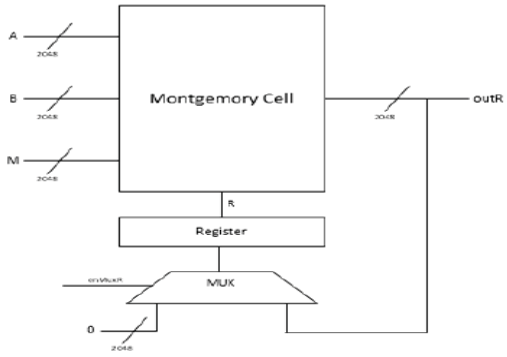

Figure -2 Montgomery Core (RTL)

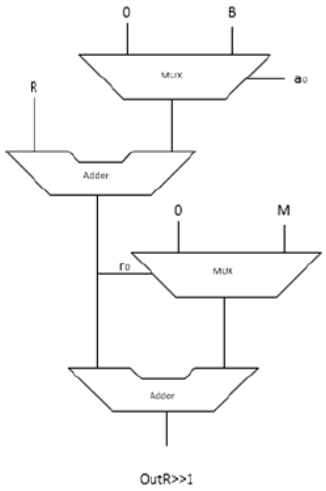

Figure -3 Montgomery Cell (RTL)

The controller generates proper control signals based on counter which will control the iteration processes. It generate control signals to show the SIPO and PISO modules I/Os. Using verilog HDL, the hardware is implemented. It requires 2048 clock cycles for generate parallel inputs to parallel output. Therefore, the time taken by the design is 2048 cycles and uses 17540 LEs , which utilizes 25.63% of total LEs in Altera DE2-70.

Advantages

For comparing the performance of this system with the other existing designs Area Time Square (AT^2) is calculated .The slices of the design is represented by LUT obtained from synthesized process. This design only use 15840 LUT and needs only 2048 clock cycles to complete the MM.

This design implementation based on AT^2 parameter, in the target FPGA which is Altera DE2-70 provides good result among other existing system.

Table 1. Performance comparison based on AT^2 parameter

Area Occupied	Total Clock cycle (T)	AT^2
15480 LUTs, 2060 Reg,17540 Les	2048 clock cycle	7.35×10^{10}

3. Montgomery multiplier for faster Cryptography

In [3] 32-bit implementation of a faster Montgomery algorithm for performing modular multiplication. Basically this method is based on the method proposed by Montgomery for modular multiplication and shows good results compared to the existing methods.

Methodology for Faster Montgomery Multiplier

For hardware implementation, reducing the chip area is the motivation behind this optimized algorithm. There is possibility to implement this method is by pre-compute four values to add the intermediate output.

Algorithm: Faster Montgomery Multiplier

Inputs: A',B',P *($0 \leq A', B' < P$) each having n-bits.*

Output: $(A'*B'(2^n)^{-1}) \bmod M$

a_i:i^{th}bit of A';

K_0: last significant bit of K;

M_0: last significant bit of M;

B'_0: last significant bit of B';

R=B'+P;

Initialize K_0= M_0= 0;

For(i=0; i<n; i++)

{

If ((K_0=M_0) && ! a_i) then I=0;

if((K_0!=M_0) && !a_i) then I=M;

if (!(K_0^M_0^B_0) && a_i) then I=Y;

if ((K_0^M_0^B_0) && a_i) then I=R;

K and M=K+M+I;

K=K>>1; M=M>>1;

Output =K+M;

If(Output $\geq$ P) then Output=Output-P;

3.1 Design Evaluation

The result shows that Faster Montgomery multiplier consumes less power and less area.

Table 2 Shows the delay analysis of MM and faster MM using Xilinx 14.2

Multiplier	Area			Delay
	slice register used	LUTs	Bonded IOBs	
Normal Montgomery Multiplier	69	270	195	41.702ns
Faster Montgomery Multiplier	81	227	131	32.55 ns

3.2 Simulation Results

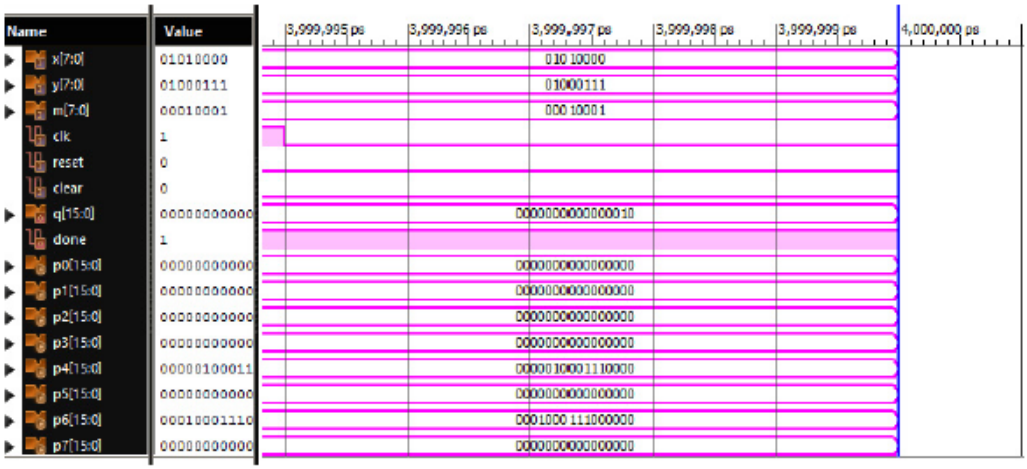

Figure 4: Simulation result for faster Montgomery Multiplier using Xilinx 14.2

Advantages

Since the possible values are already stored in a LUT, it will reduce the number of necessary additions. A simple structure method requires very less amount of computational time.Also, here power consumption is less.

4. Vedic multipliers

In [4] Vedic multiplication is used for the multiplier architecture. Basically there are three steps for Montgomery modular multiplication.

1. Converting the operands into Montgomery domain.

2. Multiplication of the operands.

3. Converting the result back into integer form.

Since the speed of the system depends on the multiplier and adders used in the system, Vedic Multiplier is used as the multiplier architecture which is considered as the faster multiplier than booth multiplier and array multiplier. In [5] a high speed 16x16 multiplier is designed by using Urdhawa Tiryagbhayam Sutra. This method helps in reduction of design architecture in the processor by generating the partial products and sums in one step. Figure 5 depicts the flow chart for multiplicative inverse which explain the below methodology.

4.1 Methodology

4.1.1 Steps

1. Conversion of integer form to Montgomery domain

*G(K)=(K*r) %N;*

*G(M)=(M*r) %N;*

2.Multiplication

*Z=G(M)*G(K)*r-1%N;*

3. Transform the result back to the integer form

*W=G-1(Z)W=Z*R-1%N;*

4.1.2 R computation

R should be in the form of 2^k and should be greater than the prime number p. Consider a variable Temp and shift right simultaneously with the prime number shift left till the MSB of the prime number becomes one. And all these operations are put inside the loop.At the end of the loop Temp holds the value of $r=2^k$.

4.1.3 Extended Euclid's algorithm can be used for calculating the inverse, where u x u^{-1}=1 mod m.

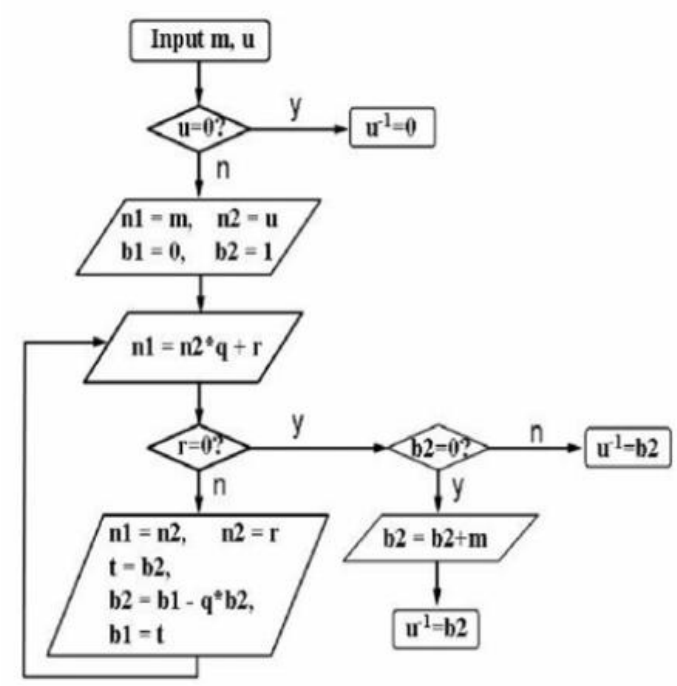

Figure 5: Flow Chart for multiplicative inverse

4.1.4 Division with the help of subtracts and shift method

The following is the steps:

1. Initialize quotient equal to zero.

2. For both the dividend and divisor align the left most digits.

3. If the left part of divisor is smaller that dividend

i. Subtract divisor with that part of the dividend.
ii. Add 1 to the LSB of dividend.

4. Repeat the above process till the number that is half the number of bits in the numerator.

5. Shift the dividend right.

6. Left part gives the quotient and right part gives the remainder.

3.2 Device Utilization

FPGA board (Virtex 6) is used for synthesis analysis:

Table 3. Device Utilization on Virtex 6 for 16 bit for exponentiation using Montgomery Modular multiplication.

Utilization	**Area occupied**	**Total Area**	**Utilization(%)**
slice registers	0	93.120	0
LUTs	33.222	46.560	71
Number of occupied slices	10.603	11,640	91
Number of bonded IOBs	79	240	32

Table 4.Device Utilization on Virtex 6 for 16 bit for Montgomery Modular multiplication.

UTILIZATION	**Area occupied**	**Total Area**	**Utilization(%)**
Slice Registers	291	126800	0
LUTs	3246	63400	5
LUT-FF pairs	218	3319	6
Bonded IOBs	65	210	30

Advantages

An efficient Multiplier block is implemented using Urdhawa Tiryagbhyam Vedic multiplier which replaces encoder multiplier.It also replaces trial division by the modulus with a number of divisions and additions by a power of 2.

5. Contributions

Here Diffie-Hellman key exchange cryptosystem for prime field have been synthesized for 512 bits in a kintex 7 board. It took 512 clock cycles to complete the process. Extra modular operations have to be done to remove the R^{-1} from the whole term which is $A.B.R^{-1}$ mod p. For removing R^{-1} from the output 512 bits register is used whose all bits are zero except the MSB bit. That is shifting one to the left side 512 times. That register is then multiplied with the output of the MMM and a modular operation is done to get A.B mod p. This implementation is proved to be efficient compared to the other methods like booth multiplier. In booth multiplier the chip area consumed is more and also the clock cycle will be more [6].

Following figures, Figure 6 and Figure 7 show the simulation results and also the synthesized results of the Montgomery modular multiplication.

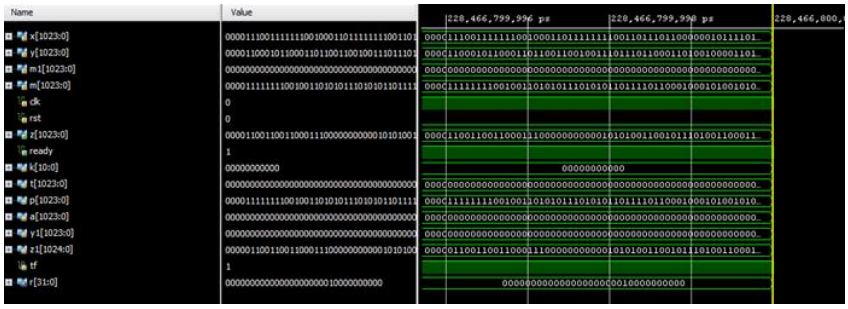

Figure 6: Simulation result for 512 bits for Montgomery modular multiplier

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	5212	0	203800	2.55
LUT as Logic	5212	0	203800	2.55
LUT as Memory	0	0	64000	0.00
Slice Registers	2072	0	407600	0.50
Register as Flip Flop	2072	0	407600	0.50
Register as Latch	0	0	407600	0.00
F7 Muxes	0	0	101900	0.00
F8 Muxes	0	0	50950	0.00

Figure 7: Utilization of the 512 bits Montgomery modular multiplier

6. Conclusion

Montgomery Modular multiplier proved to be efficient in the case of area as well as timing constraints. But one more operation of multiplication and modular operation have to be done. In the parallel operation, for every Montgomery modular multiplier there is additional operation for multiplication and modular operation, which can be avoided by pre-computing R^{n*M} *mod p* where M is the number of multiplier required and storing that value in a register. This will reduce the clock cycle as well as area in the chip.

7. References

[1] Ankush Yete, Ananya Kajava P, Hazel Melita Rodrigues, Namratha P and Kiran Kumar V.G. (2017): Implementation of Montgomery Modular Multiplication using High Speed Multiplier. International Journal of Current Engineering and Scientific Research ((IJCESR) (Online): 2394-0697, Vol: 4 (6): 99-102.

[2] Antonius P. Renardy, Nur Ahmadi, Ashbir A. Fadila, Naufal Shidqi and Tri Adiono. (2015): Hardware Implementation of Montgomery Modular Multiplication Algorithm Using Iterative Architecture. International Seminar on intelligent Technology and Its Applications, 99-102.

[3] Junfeng Fan, Kazuo Sakiyama and Ingrid Verbauwhede. (2007): Montgomery Modular Multiplication Algorithm on Multi-core Systems, IEEE Workshop on Signal Processing Systems, 261-266.

[4] NithaThampi and Meenu Elizabath Joseb. (2016): Montgomery Multiplier for Faster Cryptosystems. Science Direct Procedia Technology No, 25: 392-398. (Global Colloquium in Recent Advancement and Effectual Researches in Engineering, Science and Technology (RAEREST). www. Science.direct.com

[5] Ratna Raju, B. (2013): "A High Speed 16×16 Multiplier Based On Urdhva Tiryakbhyam Sutra". International Journal of Science Engineering and Advance Technology, IJSEAT, Vol: 1, (5) 126-132.

[6] Shinde, K. (2016): Hardware Implementation of Configurable Booth Multiplier on FPGA. International Journal of VLSI Design Communication, Vol.4 (1) 99-103.

[7] www.xilinx.com

[8] Senthilselvan, N., Udaya Sree, N., Medini, T., Subhakari Mounika, G., Subramaniyaswamy, V., Sivaramakrishnan, N., & Logesh, R. (2017). Keyword-aware recommender system based on user demographic attributes. International Journal of Mechanical Engineering and Technology, 8(8), 1466-1476.

[9] Subramaniyaswamy, V., Logesh, R., Vijayakumar, V., & Indragandhi, V. (2015). Automated Message Filtering System in Online Social Network. Procedia Computer Science, 50, 466-475.

[10] Subramaniyaswamy, V., Vijayakumar, V., Logesh, R., & Indragandhi, V. (2015). Unstructured data analysis on big data using map reduce. Procedia Computer Science, 50, 456-465.

[11] Subramaniyaswamy, V., Vijayakumar, V., Logesh, R., & Indragandhi, V. (2015). Intelligent travel recommendation system by mining attributes from community contributed photos. Procedia Computer Science, 50, 447-455.

[12] Vairavasundaram, S., & Logesh, R. (2017). Applying Semantic Relations for Automatic Topic Ontology Construction. Developments and Trends in Intelligent Technologies and Smart Systems, 48.

[13] Logesh, R., Subramaniyaswamy, V., Vijayakumar, V., Gao, X. Z., & Indragandhi, V. (2017). A hybrid quantum-induced swarm intelligence clustering for the urban trip recommendation in smart city. Future Generation Computer Systems, 83, 653-673.

[14] Subramaniyaswamy, V., & Logesh, R. (2017). Adaptive KNN based Recommender System through Mining of User Preferences. Wireless Personal Communications, 97(2), 2229-2247.

[15] Logesh, R., & Subramaniyaswamy, V. (2017). A Reliable Point of Interest Recommendation based on Trust Relevancy between Users. Wireless Personal Communications, 97(2), 2751-2780.

[16] Logesh, R., & Subramaniyaswamy, V. (2017). Learning Recency and Inferring Associations in Location Based Social Network for Emotion Induced Point-of-Interest Recommendation. Journal of Information Science & Engineering, 33(6), 1629–1647.

[17] Subramaniyaswamy, V., Logesh, R., Abejith, M., Umasankar, S., & Umamakeswari, A. (2017). Sentiment Analysis of Tweets for Estimating Criticality and Security of Events. Journal of Organizational and End User Computing (JOEUC), 29(4), 51-71.

[18] Indragandhi, V., Logesh, R., Subramaniyaswamy, V., Vijayakumar, V., Siarry, P., & Uden, L. (2018). Multi-objective optimization and energy management in renewable based AC/DC microgrid. Computers & Electrical Engineering.

[19] Subramaniyaswamy, V., Manogaran, G., Logesh, R., Vijayakumar, V., Chilamkurti, N., Malathi, D., & Senthilselvan, N. (2018). An ontology-driven personalized food recommendation in IoT-based healthcare system. The Journal of Supercomputing, 1-33.

[20] Arunkumar, S., Subramaniyaswamy, V., & Logesh, R. (2018). Hybrid Transform based Adaptive Steganography Scheme using Support Vector Machine for Cloud Storage. Cluster Computing.

[21] Indragandhi, V., Subramaniyaswamy, V., & Logesh, R. (2017). Resources, configurations, and soft computing techniques for power management and control of PV/wind hybrid system. Renewable and Sustainable Energy Reviews, 69, 129-143.

[22] Ravi, L., & Vairavasundaram, S. (2016). A collaborative location based travel recommendation system through enhanced rating prediction for the group of users. Computational intelligence and neuroscience, 2016, Article ID: 1291358.

[23] Logesh, R., Subramaniyaswamy, V., Malathi, D., Senthilselvan, N., Sasikumar, A., & Saravanan, P. (2017). Dynamic particle swarm optimization for personalized recommender system based on electroencephalography feedback. Biomedical Research, 28(13), 5646-5650.

[24] Arunkumar, S., Subramaniyaswamy, V., Karthikeyan, B., Saravanan, P., & Logesh, R. (2018). Meta-data based secret image sharing application for different sized biomedical images. Biomedical Research,29.

[25] Vairavasundaram, S., Varadharajan, V., Vairavasundaram, I., & Ravi, L. (2015). Data mining-based tag recommendation system: an overview. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 5(3), 87-112.

[26] Logesh, R., Subramaniyaswamy, V., & Vijayakumar, V. (2018). A personalised travel recommender system utilising social network profile and accurate GPS data. Electronic Government, an International Journal, 14(1), 90-113.

[27] Vijayakumar, V., Subramaniyaswamy, V., Logesh, R., & Sivapathi, A. (2018). Effective Knowledge Based Recommeder System for Tailored Multiple Point of Interest Recommendation. International Journal of Web Portals.

[28] Subramaniyaswamy, V., Logesh, R., & Indragandhi, V. (2018). Intelligent sports commentary recommendation system for individual cricket players. International Journal of Advanced Intelligence Paradigms, 10(1-2), 103-117.

[29] Indragandhi, V., Subramaniyaswamy, V., & Logesh, R. (2017). Topological review and analysis of DC-DC boost converters. Journal of Engineering Science and Technology, 12 (6), 1541–1567.

[30] Saravanan, P., Arunkumar, S., Subramaniyaswamy, V., & Logesh, R. (2017). Enhanced web caching using bloom filter for local area networks. International Journal of Mechanical Engineering and Technology, 8(8), 211-217.

[31] Arunkumar, S., Subramaniyaswamy, V., Devika, R., & Logesh, R. (2017). Generating visually meaningful encrypted image using image splitting technique. International Journal of Mechanical Engineering and Technology, 8(8), 361–368.

[32] Subramaniyaswamy, V., Logesh, R., Chandrashekhar, M., Challa, A., & Vijayakumar, V. (2017). A personalised movie recommendation system based on collaborative filtering. International Journal of High Performance Computing and Networking, 10(1-2), 54-63.