مجلة كلية التربية الاساسية
كلية التربية الاساسية – الجامعة المستنصرية
Journal of the College of Basic Education Vol.30 (NO. 124) 2024, pp. 69-86

Navigating The Cloud: Enhancing Human-Robot Collaboration In Industry 4.0 Through Digital Twins And Cloud-Based Systems

Hayder Hasan Ali
University Mustansiriyah- Directorate Student Hostels
Asst. Prof. Dr. Khalil Mershad
Modern University For Business & Science (MUBS) / College Of Science

Abstract

Industry 4.0 Is Geared Towards Establishing Sophisticated Industrial Ecosystems That Necessitate Secure And Efficient Collaboration Between Humans And Robots, Termed Human-Robot Collaboration (HRC). Within This Framework, HRC Leverages Cutting-Edge Collision Avoidance Technology Capable Of Detecting Objects And Foreseeing Possible Collisions, Thereby Calculating Alternative Pathways To Avert Any Direct Contact. Furthermore, The Concept Of Digital Twins Emerges As An Innovative Solution, Offering A Platform To Simultaneously Evaluate The Consequences Of Various Control Strategies Within A Digitally Simulated Environment. While Cloud Computing Infrastructure Can Furnish The Requisite Computational Prowess, It May Simultaneously Introduce Challenges Such As Increased Latency And Variability In Communication, Known As Jitter, Which Could Potentially Hinder System Performance. Cloud Technologies Are Renowned For Their Inventive Approaches To Ease The Workload For Software Developers And System Operators, Raising Questions About Their Integration With Digital Twin Methodologies And The Resilience Of Robotic Systems Against The Potential Delays Incurred From Cloud-Based Services. This Research Endeavors To Navigate These Complexities By Integrating A Blend Of Both Public And Private Cloud Services, Which Are Distinguished By Their Unique Parallel Processing Capabilities. This Study's Contributions Are Threefold. Firstly, It Introduces A Novel Method Designed To Gauge The Efficacy Of Diverse Strategies By Focusing On Their Associated Latency Impacts. Secondly, It Delineates A Tangible Application Of HRC, Illuminating Its Practical Benefits And Real-World Applicability. Lastly, It Defines A Crucial Performance Metric Intended To Assess The Efficiency Of These Systems Thoroughly. By Delving Into These Aspects, The Research Aims To Conduct A Comprehensive Analysis Of The Strengths And Weaknesses Inherent In Various Technological Methodologies And Their Consequential Effects On The Performance Of HRC Systems Within The Ambit Of Industry 4.0.

Through This Detailed Examination, The Study Seeks To Provide Valuable Insights Into Optimizing Human-Robot Collaboration In Industrial Settings, Ensuring Both Security And Efficiency Are Upheld

Keywords : Digital Twins, Industry 4.0, HRC, Cloud

I. Introduction

Industry 4.0 Envisions Future Production Systems That Are Versatile, Extendable, And Readily Customizable. These Systems Provide Production That Is Not Dependent On Particular Items, Use Manufacturing Procedures That Are Suitable For Several Products, And Adapt Capacity Based On Current Demand Levels. Human-Robot Collaboration (HRC) Is Essential For Future Workflows To Achieve Flexible Manufacturing Processes. HRC Integrates Human Craftsmanship And Cognitive Abilities With The Accuracy And Power Of Robots. Novel Dangers Arise From This Distinctive Link And Are Governed By Specific Safety Regulations. Regrettably, There Is A Compromise Between Safety And Productivity, Perhaps Resulting In Reduced Production Due To Existing Requirements. The Robots' Mobility Significantly Affects Production. Accurate Control Of Robot Trajectories May Enhance Production While Upholding Safety Requirements [1].

Collision Avoidance Is A Crucial Task Involving The Identification Of Possible Collisions While Adhering To A Set Route And Choosing Several Options. Digital Twins Provide An Effective Method To Assess The Effects Of Various Control Tactics In A Simulated Virtual Environment, Eliminating The Need To Physically Move Robots. These Methods Enable The Concurrent Calculation And Assessment Of Several Control Mechanisms To Analyze Their Potential Future Outcomes. Using Digital Twins May Demand Significant Computing Resources, Particularly When Several Situations Must Be Simulated Concurrently Under Tight Time Limits. Cloud Systems, Whether Public Or Private, May Provide The Necessary Computational Capacity But Can Have Increased Latency And Temporal Unpredictability. Cloud Technology Introduces Innovative Methods Like Cloud Native Programming, Microservices, And Serverless Architectures That Have The Potential To Transform Software Development And Service Delivery. Cloud Providers Can Handle Resource Scaling, Life-Cycle Management, Resilience Configuration, Fault Tolerance, And Software Component Migration, Making It Easier For Developers And Operators To Manage Their Jobs [2].

This Article Analyzes Public And Private Cloud Platforms That Provide Function As A Service (Faas) Solutions And Compares Them To Conventional Methods. Function As A Service (Faas) Is A Robust

Alternative To Local Multi-Threading And May Greatly Lessen The Workload For Software Developers, Among Other Benefits. We Have Three Main Contributions. We Provide A Measuring Tool To Assess Different Parallel Processing Systems Based On Their Latency Attributes. We Will Analyze A Practical HRC Application That Involves Calculating Several Trajectories And A Key Performance Indicator (KPI). We Assess The Advantages And Disadvantages Of Various Techniques And Their Performance Attributes [3].

Ii. Related Work And Background

This Section Provides A Summary Of Previous Studies On The Control Of Industrial Robots And Describes Computational Virtualization Techniques Intended For Use. We Provide A Brief Summary Of A Highly Relevant Public And Private Cloud Platform. The Previous Studies We Present Explain What Has Been Achieved Regarding Control, And We Explain How It Is Measured In The Next Section.

A. Industry 4.0

Industrial Applications Are Crucial For The Internet Of Things (Iot). Cloud Robotics Is A Prominent Area Of Focus In Industrial Robotics Research, Since Recent Studies Have Shown The Advantages Of Linking Robots To A Centralized Processing Unit. A) Utilizing Advanced Computing Resources In A Centralized Cloud For Machine Learning (ML) Tasks; B) Reducing The Cost Per Robot By Transferring Functions To A Central Cloud For Sharing; C) Enhancing The Integration Of External Sensor Data And Enhancing Collaboration With Other Robots And Machinery; D) Enhancing Function Reliability By Running Multiple Instances As Hot Standby In The Cloud, Allowing For Immediate Takeover Of Operations From A Faulty Primary Function Without Interruption [4].

Industrial Robotics Research Places Significant Emphasis On Human-Robot Collaboration (HRC). Reference [5] Provides An Overview Of The Many Types Of Human-Machine Collaboration In Assembly And The Technologies That Facilitate Human-Robot Cooperation. The Authors Cited In Reference [6] Used A Machine Learning Approach To Allow Industrial Robots To Navigate Obstacles Or People In The Workplace. This Is Mostly Due To The Wide Range Of Potential Occurrences That Might Take Place In A Disorganized Setting. Representing The Changing Working Environment And Human Motions Deterministically Is Challenging. This Task Necessitates The Use Of Machine Learning To Rapidly Detect And Separate Items...

B. Novel Cloud Technologies

Effective Resource Allocation And Coordination Procedures Are Essential For Arranging Virtual Control Functions Within Specified Latency Limits In The Cloud For Applications Requiring Low Latency. Internal Operations Of The Cloud Platform Might Impact The Overall Latency. In A Prior Research Project, We Conducted A Thorough Analysis Of Amazon's Public Cloud Platform (AWS), Specifically Examining Factors That Contribute To Delays And Affect The Performance Of Latency-Sensitive Applications In This Setting. Choosing Cloud Services Strategically During The Design Phase And Ensuring The Application Can Withstand Delays Of Around 100 Milliseconds Allows For The Feasibility Of The Cloud-Native Approach, Along With Its Benefits. Edge And Fog Computing Are Innovative Technologies That Improve Standard Cloud Computing By Placing Computer Resources In Closer Proximity To Users And End Devices. Several Studies Examine The Potential Uses, Required Technology, And Coordination Choices Of Edge Computing. The Key Aspect Of The Edge In Industry 4.0 Applications Is Its Capacity To Quickly Meet Rigorous Schedule Requirements, Even In The Face Of Delays From Internal Operations [7].

C. Amazon’s AWS Platform

AWS, The Leading Public Cloud Provider, Provides A Range Of Choices To Meet The Processing Requirements Of Industry 4.0 Applications. It Provides Three Services That Enhance Computational Performance And Provide Exceptional Scalability. EC2 Provides Rapid Virtual Machine Deployment On Amazon's Infrastructure, Offering A Higher Level Of Service. Lambda Is A Serverless Platform That Hides The Underlying Infrastructure, Whereas Elastic Container Service (ECS) Enables Customers To Install Docker Containers On EC2 Servers [8]...

Users Mostly Handle Resource Management. These Choices Differ Significantly In Cost And Configuration. EC2 Offers A Variety Of Preconfigured Packages Called Flavors, Which Include Virtual CPU, Memory Capacity, And Network Speed. It Allows The Operation Of Many Applications Without Stringent Limitations. Users May Choose Virtual Machines With A Maximum Of 128 Virtual Cpus, Enabling Substantial Parallelization At The Instance Level. ECS Allows Programs To Operate Across Several Platforms, But Its Instances Are Limited To Accessing A Certain Range Of Resources. Containers May Be Allocated Up To Four Virtual Cpus At Most. Running Many Instances Simultaneously May Address This Issue, But Configuring Them Might Be Time-Consuming.

Using Lambda Simplifies Code Scalability And Parallel Execution, But It Imposes Restrictions On The Kinds Of Code That Can Be Run And The Configuration Choices Users May Make. It Is Restricted To Certain Runtimes And Distributes Compute Resources According To Memory Configurations In An Arbitrary Manner. Lambda Relies On Event Sources Provided By AWS To Trigger A Function, While EC2 And ECS Instances May Be Accessed In Any Manner Chosen By The User. The Platform Generates New Instances Automatically When There Are None Available To Handle A Request, Up To A Set Maximum Limit. EC2 And ECS Charges Are Determined By The VM And Container Uptime, Whereas Lambda Pricing Is Based On Execution Time, Allotted Resources, And Request Count [10,11]...

D. Openwhisk

Openwhisk Is An Open-Source Function As A Service (Faas) Platform That Offers Several Deployment Options And A Distributed Serverless Cloud Solution. The Most Fundamental And Advisable Approach Is To Use The Kubernetes Container Management System, Which Accommodates Several Software Container Frameworks. Openwhisk Adheres To The Faas Programming Style, Allowing Developers To Create Actions, Which Are Functional Logic, In Many Supported Languages [12]. The Framework Conducts These Actions In Response To Certain Events Or Triggers, Regardless Of Their Size. Compared To A Virtual Machine Or A Complete Container, A Scale Unit Is A Software Feature That May Have A Smaller Footprint. The Platform Oversees The Whole Infrastructure, Including Both Physical And Virtual Servers, And Adjusts Docker Containers In Real-Time Based On Changes In The System's Requirements [13,14]..

Iii. Human-Robot Collaboration

This Section Emphasizes Key Elements That Might Assist In Creating Digital Twins And Presents A Notable Potential Application That Motivates Our Research. A Collaborative Robot, Also Known As A Cobot, Is A Kind Of Robot Hardware Specifically Created To Provide Safety In An Environment Where Human-Robot Cooperation (HRC) Occurs. Reference [5] Explains That Cobots May Switch Between The Following States During An Emergency: Emergency Stop, Safeguard Stop, Protective Stop, Idle, And Disconnected. Although These States Vary, They All Have The Characteristic That The Cobot Does Not Participate In Any Productive Activity In Any Of Them. Decrease The Time Spent In Certain States To Improve Productivity Metrics [15]..

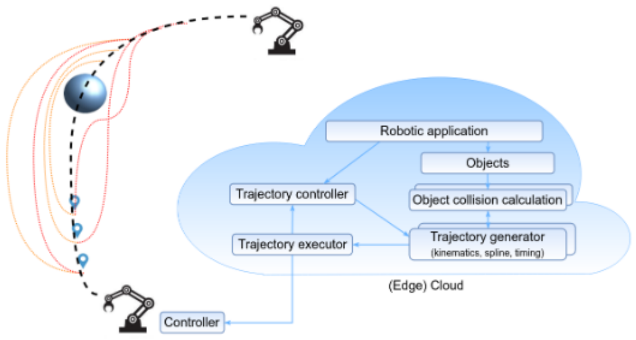


Figure 1 Depicts A Use-Case Scenario Of Human-Robot Collaboration..

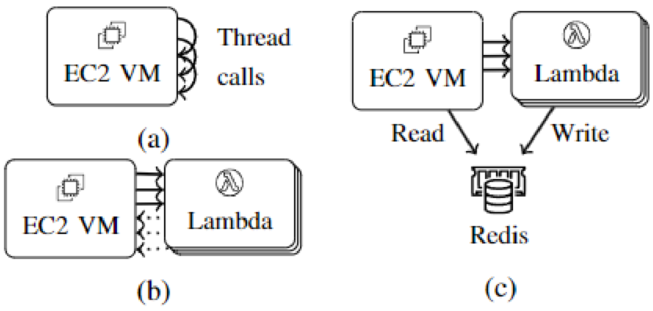


Figure 2 Displays Targeted Scenarios: Single VM (A), Synchronous (B), And Asynchronous Lambda Calls (C).

In The Future, Advanced Collaborative Robots Might Accurately Plan Their Paths To Avoid Accidents And Ensure Optimal Efficiency In Scenarios Where Humans And Robots Work Together [16]. Figure 1 Illustrates A Theoretical Future Situation Including The Use Of HRC. In A Complicated Manufacturing Setting Known As A Robotic Cell, A Sophisticated Robotic Program Controls Conveyor Belts For Transporting Workpieces, People, Or Cargo, A Robotic Arm, And Other Parts. We Are Interested In Managing The Robotic Arm [17]. The Primary Program Utilizes A Trajectory Controller To Guide The Arm To A Specified Location By Invoking A Trajectory Generator To Establish The Optimal Path. The Trajectory Of The Robot May Be Influenced By Various Limits And Goals Via Cooperation Between The Trajectory Executor And The Low-Level Controller. Reassessing The Current Approach May Be Required Considering The Cell's Interactive Behavior And Its Contacts With Human Persons. The Arm Should Navigate Around Obstacles Instead Of Stopping The Robot. The Complexity Of The

Cell Will Dictate The Computational Workload Required For These Computations. Multiple Contingency Routes Should Be Planned Ahead Of Time And Readily Available In Changing Situations [18,19]..
Multiple Alternative Routes Are Computed From Different Starting Points Along The Current Trajectory Upon Detecting A Likely Collision. Figure 1 Displays Three Initial Locations Marked With POI Icons. A Variety Of Options With Distinct Characteristics Are Calculated From A Set Initial Location, And A Sophisticated Algorithm Is Used To Assess The Probability Of Item Collisions. Execute The Second Calculation For Each Segment Of The Authorized Path, Eliminating Any Duplications. Calculating Trajectories And Collisions Requires Several Instances Of These Processes To Operate Simultaneously[20]. The Quality Of Diversions And The Platform's Response Time Will Significantly Affect The Cell's Overall Performance, Particularly When Detours Are Easily Available. If A Detour Is Provided From The First Designated Location To The Second Indicated Area, The Robot Is Unable To Follow The Prescribed Path. If The Latency Requirements Are Met, Certain Components Of The Robot's Control System Might Be Operated In Cloud Computing Settings [22]..

Methodology:

- Combination Of Cloud Services: Employing A Mix Of Public And Private Cloud Services To Explore Different Parallel Processing Techniques, Aiming To Optimize Performance While Mitigating Latency Issues.
- Measurement Technique: Developing A Measurement Method To Evaluate The Effectiveness Of Various Strategies Based On Latency, Providing A Quantitative Basis For Comparison.
- Practical Application: Demonstrating The Practical Application Of HRC In An Industrial Setting, Including The Deployment Of Advanced Collision Avoidance Systems.
- Performance Metrics: Establishing Significant Performance Metrics To Assess The Effectiveness Of HRC Systems In Real-World Scenarios.

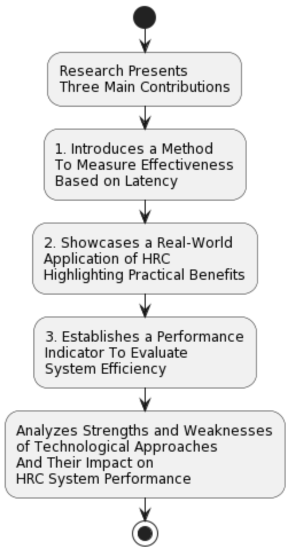

Research Presents Three Main Contributions
1. Introduces a Method To Measure Effectiveness Based on Latency
2. Showcases a Real-World Application of HRC Highlighting Practical Benefits
3. Establishes a Performance Indicator To Evaluate System Efficiency
Analyzes Strengths and Weaknesses of Technological Approaches And Their Impact on HRC System Performance

Figure 3 The Three Main Contributions

Iv. Measurement Methodology

This Section Focuses On The Measuring Approach We Used To Analyze Various Deployment Choices For Comparable Applications And Program Patterns.

A. Simple Test Functions

Initially, We Use Hypothetical And Resource-Intensive Techniques For Preliminary Testing. Figure 2 Displays Three Distinct Setups That Were Developed To Evaluate The Effectiveness Of Parallelization With Function As A Service (Faas). We Examined A Scenario In Which Identical Operations Must Be Executed, But They Commence From Varying Initial Conditions. We Use Go Code To Run Resource-Intensive Tests And Gauge The Code's Execution Duration....
We Use A M5.24xlarge AWS EC2 Instance With 768 Gib Of RAM And 96 Vcpus With Basic Multi-Threading As Our Starting Point. Figure 3 Shows The Test Code Initiating Several Threads And Then Waiting For Each Thread To Complete Its Computations. Figure 2b Displays The Configuration For The Second Test Scenario, Using AWS Lambda For Concurrent Processing. Once The EC2 Instance Creates Threads, These Threads Then Invoke Lambda Functions Synchronously. While The Caller Threads Are In A Waiting State, The Lambda Functions Do The Specified Computations And Provide The Results To The EC2 Virtual Machine. We Can Calculate The Overall Execution Time In Both Scenarios By Subtracting

The Start Time Of The First Thread From The Completion Time Of The Final Thread, As Seen In Figure 3. Figure 2c Depicts A Situation Where Lambda Functions Are Invoked Asynchronously By Threads Initiated By The EC2 Instance. The Outputs Of These Functions Are Stored In An In-Memory Redis Cache. Every Function Saves Its Result With A Distinct Key Provided By The EC2 Master. The EC2 Instance Consistently Retrieves The Output Of Each Lambda Function From The Cache. Execution Time In This Arrangement Is Defined As The Duration From The First Call To The First Lambda Function Until The EC2 Instance Can Access Each Provided Key. We Assess Situations On Our Openwhisk Platform By Using Actions For Both Synchronous And Asynchronous Lambda Calls, Rather Than Lambdas. We Implemented The Openwhisk Environment Using Kubernetes Following The Prescribed Approach. The AWS EC2 Instance Has A Total Of 96 Virtual Cpus..

B. Real Use-Case: Trajectory Calculation

Here, We Are Examining A Smaller Version Of The Use-Case Described In Section III. The Software Aims To Provide Alternative Paths To A Mobile Robot Arm To Prevent Collisions With Unforeseen Objects. Allocating The Program With The Appropriate Computing Capacity Will Enable It To Generate Several Trajectories For Chosen Places Ahead Of The Arm Reaching Them. The Elements Shown In Figure 5 Constitute The Custom Designed Software That Integrates Certain Functionalities. A Controller Supervises Jobs Near The Robot Arm And Interfaces With A C++ API To Get Longer Trajectories. Figure 1 Illustrates The C++ API, Which Is A Locally-Operated Multi-Module System. It Is Responsible For Sending Requests To The Cloud For Assistance And Receiving Responses From Several Local Controllers To Support Many Robots. We Established An EC2 Virtual Machine Instance In The Cloud To Execute Simultaneous Lambda Function Calls Following The Steps Shown In Figure 2b. Lambda Functions Are Used To Calculate Object Collisions, Unlike Other Functions Which Are Executed Locally On The Virtual Machine. The Important Performance Metric Is The Quantity Of Beneficial Trajectories Computed By Our Program Within A Certain Timeframe..

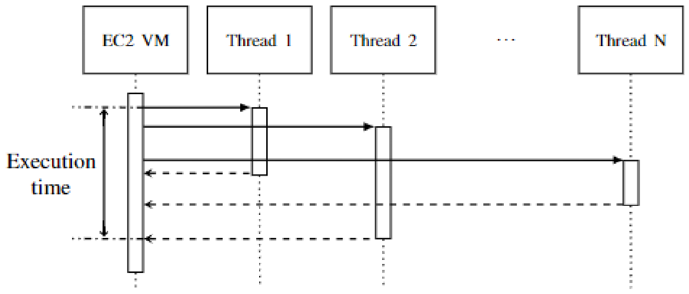


Figure 3 Illustrates Parallel Execution Via The Use Of Several Threads..

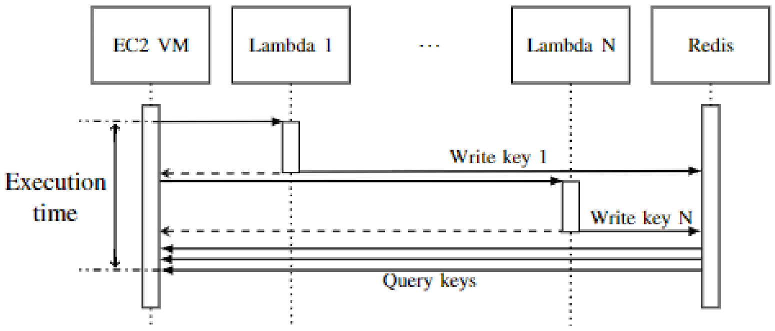


Figure 4 Illustrates Parallel Execution With Asynchronous Lambda Calls.

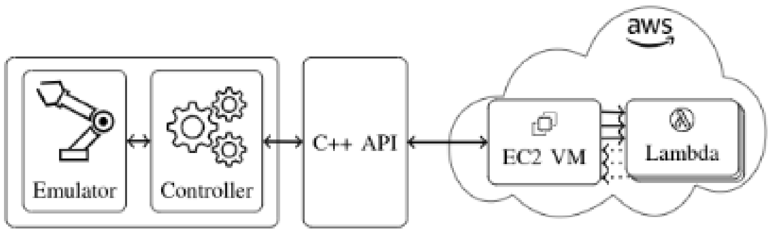


Figure 5 Displays Our Collision Avoidance Measuring Setup.

V. Experimental Evaluation

This Section Presents The Primary Findings Derived From A Thorough Performance Study Conducted Using Our Measuring Methodology..

A. Results With Simple Test Functions

We Used The Conventional Go Test Function To Do Many Measurements. The Tests Are Created To Compare And Evaluate The Efficiency Of Several Systems Running Simultaneously. We Specifically Assess The Performance Improvement Of Function As A Service (Faas) Systems With Either Unlimited Or Restricted Virtual Cpus Compared To The Additional Workload Caused By Background Activities. Various Test Functions With Varying Computing Complexities Are Assessed To Determine Their Impact

On Overall Performance. Table I Shows The Performance Timings Of Certain Synthetic Functions Evaluated On A Standard CPU Core. We Contend That A Crucial Factor Affecting Total Performance Is The Quantity Of Parallel Workers, Such As Threads, Lambda Functions, Or Actions. We Used 10, 100, 200, 500, And 1000 Parallel Workers Sequentially For Process Analysis. Figure 6a Displays The Average Execution Times Of Different Workers On All Platforms. The Function's Complexity Directly Impacts The Execution Time On Both AWS And Openwhisk. Due To AWS Providing A Limitless Amount Of Cores, The Desired Outcome Is Attained. Openwhisk Maintains Its Speed On A Virtual Machine With 96 Virtual Cpus Even While Installing 1000 Activities. This Circumstance Happens When Actions Are Placed In A Queue Before Being Scheduled. Even With 10,000 Concurrent Activities Allowed In The Openwhisk Configuration, Operations May Still Experience Significant Wait Time. The Multi-Threaded Version Of The Test Software Shows Decreased Performance When Using All 96 Available Cores, As Anticipated...

Secondly, Figure 6b Displays The Total Execution Time For Several Situations Involving One Thousand Workers. Two Reference Measurements Are Provided For Multi-Threaded Processes: One Using A Big Virtual Machine (VM) With 96 Cores And The Other Using A Smaller VM With 8 Cores. An Important Distinction Is Expected. AWS Lambda Calls, Whether Synchronous Or Asynchronous, Provide Outstanding Performance For Each Specified Scenario. When Dealing With Huge Functions, AWS Solutions Are More Effective Than A Multi-Threaded Program Running On A Large Virtual Machine. The Medium-Sized Virtual Machine (VM) Routinely Exhibits Worse Performance Compared To AWS. The Workers' Complexity Greatly Influences The Outcomes Of Openwhisk. The Platform Overhead And Queueing Delay Are Caused By Several Factors, Such As Internal Scheduling And Docker In Docker. With 1000 Workers, Synchronous And Asynchronous Modes Provide Comparable Performance Overall..

Figure 7 Illustrates The Performance Parameters Based On The Number Of Staff. The Findings Demonstrate The Comparison Between Medium-Sized Workers Using Synchronous (Fig. 7a) And Asynchronous (Fig. 7b) Calls, Using The Multi-Threaded Operation As A Reference Point. Openwhisk Works In Two Methods, With AWS Synchronous Calls Being Somewhat Faster Than The Asynchronous Technique. Openwhisk Demonstrates Superior Performance Compared To AWS With A Smaller Number Of Workers (10), Suggesting That AWS May Have A Greater Platform

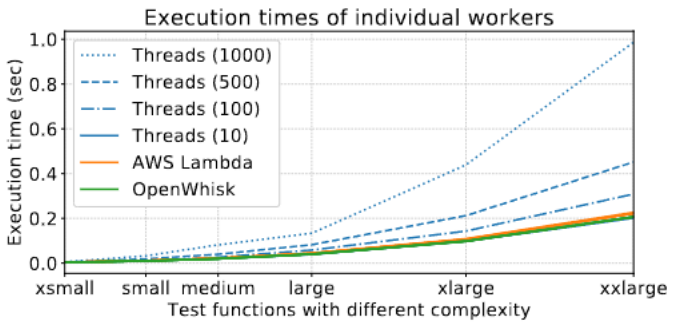
Overhead. The Total Performance Decreases As The Number Of Workers Equals Or Exceeds The Number Of Underlying Cores In Openwhisk, As Anticipated. Openwhisk's Scalability Remains Stable Even As The Number Of Workers Rises, Unlike AWS. During Our AWS Trials, You Have The Freedom To Use Unlimited Cores And Lambda Functions Simultaneously. The Findings Validate That Openwhisk Is Effective As A Function As A Service (Faas) Platform In Private Edge Cloud Deployments...

B. Results With Trajectory Calculation

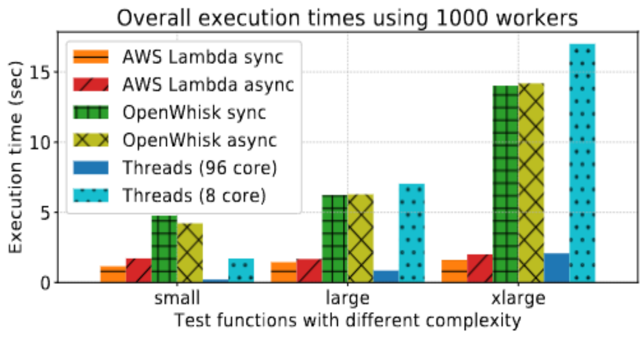
We Incorporated The Pertinent Sections Of The HRC Use-Case From Section III Into A C++ Program Designed For Use With Universal Robots' UR5 Industrial Robot Arm. We Adhered To The Technique Outlined In Figure 5 Of Section IV-B Throughout The Studies. We Assessed Several Methods For Simultaneous Object Collision Detection For Each Segment Of The Trajectory Being Studied. This Key Aspect Directly Impacts The Number Of Beneficial Alternative Paths That May Be Shown To The Robot Within A Certain Timeframe. Figures 8a And 8b Show The Robotic Arm's Movement In Cartesian Space While Operating On Our Large Virtual Machine Hosted On AWS, For Single-Threaded And Multi-Threaded Implementations, Respectively. The Top View Of Our GUI Displays Red Curves In The Graphs Representing The X-Y Movement Of The Arm's Termination. This Deployment Might Provide More Efficient Alternative Routes With Fewer Accidental Deviations. Our Solution Transitioned To The AWS Lambda Framework, But Saw A Significant Decrease In Performance Because Of The Use Of The Older C++ Runtime (V0.1.0). Substantial Delays, As Seen In Figure 8c, Led To Operational Failure...

Table I Displays The Complexity Of The Functions In Reference Measures.

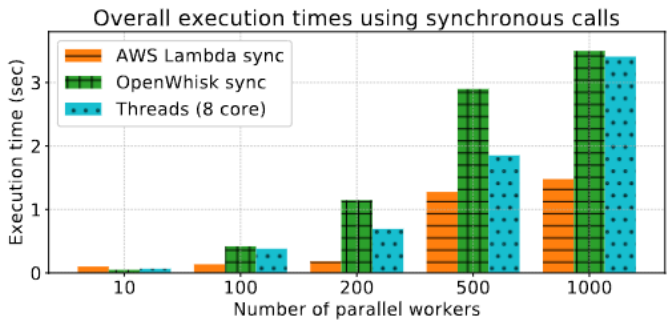
Label	Execution Time
Xsmall	~5 Msec
Small	~10 Msec
Medium	~20 Msec
Large	~50 Msec
Xlarge	~100 Msec
Xxlarge	~200 Msec



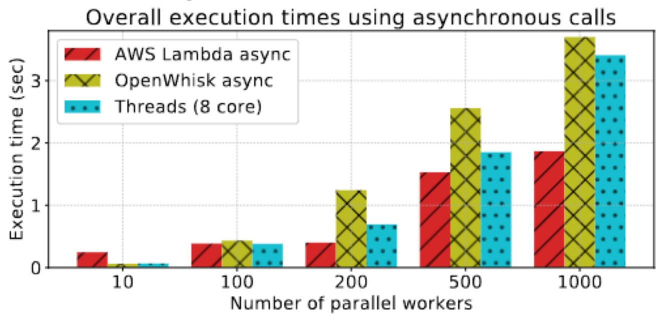
(A) Mean Execution Duration Of Workers With Varying Levels Of Complexity



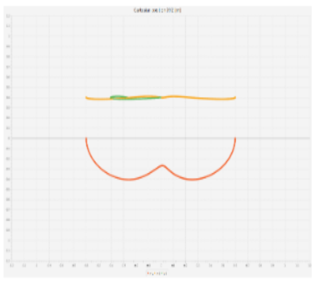
(B) Total Execution Time Of Jobs With 1000 Concurrent Workers Figure 6 Displays The Execution Times Per Worker And Total On Various Platforms..

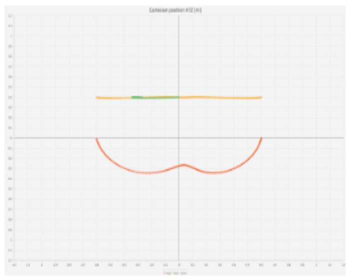


(A) Synchronous AWS Lambda Invocations, Synchronous Openwhisk Functions, Multi-Threading On A Virtual Machine With 96 Virtual Cpus.

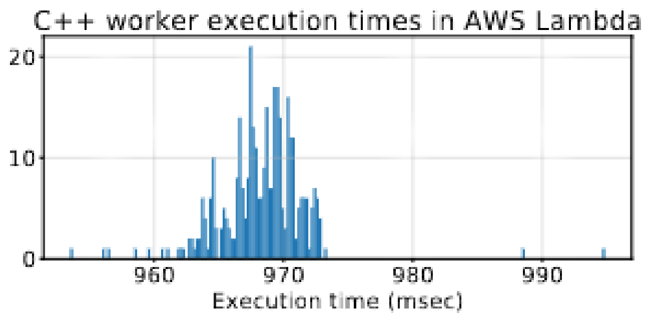


(B) Asynchronous AWS Lambda Calls, Asynchronous Openwhisk Actions, Multi-Threading On A VM With 96 Vcpus (As A Reference).
Figure 7 Displays The Overall Performance Across Several Platforms When Using Medium-Sized Personnel.





(A) Single-Threaded Implementation. (B) Implementation Using Multiple Threads



(C) Lambda Execution Times Histogram
Figure 8 Displays The Paths Of The Robot Arm In Cartesian Space In Panels (A) And (B), Together With A Histogram Of A C++ Lambda Function On AWS In Panel (C)..

Vi. Conclusion

The Latest Virtualization Technologies Are Categorized Under The Faas Umbrella And Provide A Serverless-Like, Lightweight On-Demand Computing Solution. This Cloud Computing Service May Be Used To Operate Application Components Required In Large Quantities Simultaneously, Yet Seldom. Code May Be Efficiently Run Concurrently Using Cloud Computing, Particularly When Dealing With Fluctuating Scales. Virtualization Solutions Often Provide Additional Latency, Making Them Unsuitable For Applications And Cloud Services That Need Low Latency. We Selected An Internet Of Things Application Focused On The Management Of Industrial Robots For Our Investigation. The Digital Twin In Human-Robot Collaboration Calculates Potential Movement Paths

Simultaneously To Provide Timely Control Choices For Avoiding Collisions. We Analyzed The Duration Required To Complete The Task For These Computations In Three Unique Implementation Situations. Openwhisk Operates On A Potentially Private Cloud, AWS Lambda Functions In A Public Cloud, And A Conventional Multi-Threaded Application...

We Measured The Task's Execution Time And Compared It Across Three Scenarios With Different Degrees Of Parallelization And Job Complexity. Openwhisk Offers A Decent Balance Between Allotted Resources And Execution Speed, While AWS Lambda Gives Outstanding Performance. The Latter Outperforms Native Multi-Threaded Apps On A Mid-Range Server With 8 Cores, Particularly When Scaled Up. We Believe That A Crucial Part Of The Study That Was Overlooked Was The Resource That Was Supplied. Openwhisk May Be The Most Cost-Effective, High-Performing, And Scalable Option. Performance Analysis Enables Us To Address The Intriguing Issue Presented In The Paper's Title. Cloud Deployment Is No Longer Feasible For Upcoming Digital Twin Applications Requiring Millisecond Decision-Making Capabilities. AWS Lambda, The Highest-Performing Faas Platform In Our Evaluations, Was Used In Our C++ Solution But Failed To Meet The Necessary Fast Completion Time..

In The Future, We Want To Build The Application Using Go To Take Advantage Of Improved Support In Faas Platforms And Reduce End-To-End Latency By Using Private Edge Clouds. Aligning The Amount Of Parallelization With The Number Of Computing Cores Is Essential For Maximizing Openwhisk's Performance On The Edge..

References

[1.] Leng, Jiewu, Et Al. "Digital Twins-Based Smart Manufacturing System Design In Industry 4.0: A Review." Journal Of Manufacturing Systems 60 (2021): 119-137.
[2.] Juarez, Maria G., Vicente J. Botti, And Adriana S. Giret. "Digital Twins: Review And Challenges." Journal Of Computing And Information Science In Engineering 21.3 (2021): 030802.
[3.] Mylonas, Georgios, Et Al. "Digital Twins From Smart Manufacturing To Smart Cities: A Survey." Ieee Access 9 (2021): 143222-143249.
[4.] Apache Openwhisk. Https://Openwhisk.Apache.Org.
[5.] Docker: A Better Way To Build Apps. Https://Www.Docker.Com.
[6.] Kubernetes: Production-Grade Container Orchestration. Https://Kubernetes.Io.
[7.] Serverless: The Serverless Application Framework Powered By AWS Lambda, API Gateway, And More. Https://Serverless.Com/.
[8.] The Top 5 Cobot Kpis. Https://Voelker-Controls.Com/Wp-Content/Uploads/2019/01/UR_Robotiq__Ebook_5-Cobot_Kpis-.Pdf. Accessed: 2019-05-10.
[9.] K. Dröder, P. Bobka, T. Germann, F. Gabriel, And F. Dietrich. A Machine Learning-Enhanced Digital Twin Approach For Human-Robot-Collaboration. Procedia CIRP, 76:187 – 192, 2018.
[10.][7] Y. Guo, X. Hu, B. Hu, J. Cheng, M. Zhou, And R. Y. K. Kwok. Mo-Bile Cyber Physical Systems: Current Challenges And Future Networking Applications. IEEE Access, 6:12360–12368, 2018.
[11.] B. Kehoe, S. Patil, P. Abbeel, And K. Goldberg. A Survey Of Research On Cloud Robotics And Automation. IEEE Transactions On Automation Science And Engineering, 12(2):398–409, 2015.
[12.] J. Krüger, T. Lien, And A. Verl. Cooperation Of Human And Machines In Assembly Lines. CIRP Annals, 58(2):628 – 646, 2009.
[13.] P. Mach And Z. Becvar. Mobile Edge Computing: A Survey On Archi-Tecture And Computation Offloading. IEEE Communications Surveys And Tutorials, 19(3):1628–1656, 2017.
[14.] B. Németh, M. Szalay, J. Dóka, M. Rost, S. Schmid, L. Toka, And B. Sonkoly. Fast And Efficient Network Service Embedding Method With Adaptive Offloading To The Edge. In IEEE INFOCOM WKSHPS, 2018.
[15.] J. Pan And J. Mcelhannon. Future Edge Cloud And Edge Computing For Internet Of Things Applications. IEEE Internet Of Things Journal, 5(1):439–449, 2018.

[16.] I. Pelle, J. Czentye, J. Dóka, And B. Sonkoly. Towards Latency Sensitive Cloud Native Applications: A Performance Study On AWS. In IEEE CLOUD, 2019.
[17.] W. Shi, J. Cao, Q. Zhang, Y. Li, And L. Xu. Edge Computing: Vision And Challenges. IEEE Internet Of Things Journal, 3(5):637–646, 2016.
[18.] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, And D. Sabella. On Multi-Access Edge Computing: A Survey Of The Emerging 5G Network Edge Cloud Architecture And Orchestration. IEEE Communications Surveys Tutorials, 19(3):1657–1681, 2017.
[19.] Pylianidis, Christos, Sjoukje Osinga, And Ioannis N. Athanasiadis. "Introducing Digital Twins To Agriculture." Computers And Electronics In Agriculture 184 (2021): 105942..
[20.] Ketzler, Bernd, Et Al. "Digital Twins For Cities: A State Of The Art Review." Built Environment 46.4 (2020): 547-573.
[21.] Segovia, Mariana, And Joaquin Garcia-Alfaro. "Design, Modeling And Implementation Of Digital Twins." Sensors 22.14 (2022): 5396.
[22.] Mihai, Stefan, Et Al. "Digital Twins: A Survey On Enabling Technologies, Challenges, Trends And Future Prospects." IEEE Communications Surveys & Tutorials 24.4 (2022): 2255-2291.

مجلة كلية التربية الاساسية
كلية التربية الاساسية – الجامعة المستنصرية

Journal of the College of Basic Education Vol.30 (NO. 124) 2024, pp. 69-86

التنقل في السحابة: تعزيز التعاون بين الإنسان والروبوت في الصناعة 4.0 من خلال التوائم الرقمية والأنظمة المستندة إلى السحابة

حيدر حسن علي
الجامعة المستنصرية - قسم شؤون الاقسام الداخلية
ا.م.د. خليل مرشد
الجامعة الحديثة للادارة والعلوم (MUBS)

مستخلص البحث:

تهدف الصناعة 4.0 إلى إنشاء أنظمة بيئية صناعية متطورة تتطلب تعاونًا آمنًا وفعالًا بين البشر والروبوتات، وهو ما يسمى التعاون بين الإنسان والروبوت .(HRC) وفي هذا الإطار، يستفيد مركز HRCمن أحدث تقنيات تجنب الاصطدام القادرة على اكتشاف الأجسام والتنبؤ بالاصطدامات المحتملة، وبالتالي حساب مسارات بديلة لتجنب أي اتصال مباشر. علاوة على ذلك، يظهر مفهوم التوائم الرقمية كحل مبتكر، حيث يوفر منصة لتقييم عواقب استراتيجيات التحكم المختلفة في بيئة محاكاة رقمية في وقت واحد. في حين أن البنية التحتية للحوسبة السحابية يمكن أن توفر البراعة الحسابية المطلوبة، فإنها قد تقدم في الوقت نفسه تحديات مثل زيادة زمن الوصول والتقلب في الاتصالات، المعروفة باسم الارتعاش، والتي يمكن أن تعيق أداء النظام.

تشتهر التقنيات السحابية بأساليبها المبتكرة لتخفيف عبء العمل عن مطوري البرامج ومشغلي الأنظمة، مما يثير تساؤلات حول تكاملها مع منهجيات التوأم الرقمي ومرونة الأنظمة الآلية في مواجهة التأخير المحتمل الناتج عن الخدمات المستندة إلى السحابة. ويسعى هذا البحث إلى التغلب على هذه التعقيدات من خلال دمج مزيج من الخدمات السحابية العامة والخاصة، والتي تتميز بقدرات المعالجة المتوازية الفريدة. مساهمات هذه الدراسة هي ثلاثة أضعاف. أولاً، يقدم طريقة جديدة مصممة لقياس مدى فعالية الاستراتيجيات المتنوعة من خلال التركيز على تأثيرات الكمون المرتبطة بها. ثانيًا، فهو يحدد تطبيقًا ملموسًا لمجلس حقوق الإنسان، ويسلط الضوء على فوائده العملية وقابلية تطبيقه في العالم الحقيقي. وأخيرًا، فهو يحدد مقياس أداء حاسمًا يهدف إلى تقييم كفاءة هذه الأنظمة بدقة. ومن خلال الخوض في هذه الجوانب، يهدف البحث إلى إجراء تحليل شامل لنقاط القوة والضعف الكامنة في المنهجيات التكنولوجية المختلفة وتأثيراتها المترتبة على أداء أنظمة HRC ضمن نطاق الصناعة 4.0. ومن خلال هذا الفحص التفصيلي، تسعى الدراسة إلى تقديم رؤى قيمة حول تحسين التعاون بين الإنسان والروبوت في البيئات الصناعية، مما يضمن الحفاظ على الأمن والكفاءة..

الكلمات الرئيسية : التوائم الرقمية ، الصناعة 4.0 ، HRC ، السحابة.