A BIG DATA DRIVEN SYSTEM TO IMPROVE RESIDENTIAL IRRIGATION EFFICIENCY USING MACHINE LEARNING AND AI

Kai Segimoto[1], Nelly Segimoto[1] and Yu Sun[2]

[1]Arcadia High School, 180 Campus Dr, Arcadia, CA 91006
[2]California State Polytechnic University, Pomona,
CA, 91768, Irvine, CA 92620

ABSTRACT

California has been prone to drought; starting in 2011, there were 376 consecutive weeks of drought [1]. More effective tools are necessary to combat water scarcity, in particular in irrigation systems [3]. This paper designs an application to modify current water-saving techniques to create a more environmentally friendly irrigation system [2]. We developed a Big Data Driven System to Improve Residential Irrigation Efficiency. Our design uses the raspberry Pi controller based on an IoT system with a database connected to the cloud. We designed a mobile app to interact with the system and collect the data and a machine learning algorithm to analyze and generate recommendations based on the given data [4]. We applied our application to the irrigation systems of California Residents and conducted a qualitative evaluation of the approach. The results show that trend-based water saving techniques were effective in reducing water usage without sacrificing the health of the plants being irrigated.

KEYWORDS

Data mining, Cloud computing, Machine Learning, IoT system.

1. INTRODUCTION

For centuries, California weather records have documented intense droughts. With the further commercial and agricultural development, water scarcity has become ever prevalent and increasingly damaging to the environment as well as well-being of residents [5]. As life-long California residents, we have seen the environmental consequences of these droughts and the ineffectiveness of our water-saving techniques. Since all signs indicate that drought will persist, finding ways to reduce water usage is imperative. One of the most inefficient usages of water can be seen in residential irrigation systems. Because of the large consumption of water in irrigation systems, in every city in California, utility companies have urged residents to limit water usage by issuing mandates limiting the dates and times residents are permitted to water the plants and fining those who fail to adhere to these rules. Unfortunately, this method isn't effective -- especially since most water usage goes to irrigating lawns, and gardens in residential homes need to be watered based on weather conditions. It is also unfortunate that California residents have been slow to shift to more drought-tolerant gardens. This has led to the importation of water from nearby states -- a costly and environmentally damaging practice. In recent years, technological developments have led to the creation of irrigation systems that can be controlled virtually [6]. With the creation and adoption of irrigation systems that irrigate based on the needs of plants, California residents will be able to minimize water usage by utilizing it as efficiently as possible, thus conserving water and benefiting not only the environment, but also their finances.

David C. Wyld et al. (Eds): SIPP, NLPCL, BIGML, SOEN, AISC, NCWMC, CCSIT - 2022
pp. 101-111, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121308

In California, there are many pre-established water-saving techniques. Many tend to vary on the location of the city and the people in charge of water distribution, but in general, most residential neighborhoods set in place guidelines dictating when it is permissible to water plants [7]. These times tend to be two or three times a week, during hours of the day generally ranging from 8pm to 8am, or after dark and before sunrise. Water companies charge residents fees for not abiding to these guidelines, but this method of water regulation is hardly beneficial to reducing the amount of water usage. While initially, these guidelines may seem effective, the issue with this method is that it doesn't take into account weather conditions nor how long plants are being watered. First, it is ineffective because people already water their plants every other day, so regulating which days those are does not reduce water usage. Second, a set frequency during all seasons is not effective due to heat and precipitation being large factors in the amount of water a plant needs, and third, the time frame is so wide that many still end up over irrigating their plants due to lack of knowledge on how long their plants should be watered for. Modernized sprinkler systems are often left unadjusted to match weather conditions because the way they are constructed is not user friendly, leading to many older adults (usually the ones who can afford large, residential homes) being unable to adjust their sprinkler systems to current weather conditions. This is detrimental because sprinkler settings may be set matching summer weather, leading to an extreme use of water during winter periods that need virtually no water. Other smart sprinkler systems run into many of the same issues because individuals are unaware of how long they need to water their plants and lawns, and although they can easily control their sprinklers virtually, the lack of this knowledge leads to over-watered lawns and plants and thus a waste of valuable water [8].

In this paper, our goal is to create an irrigation system driven by data mining and machine learning algorithms to effectively control the sprinkles to save the water [9]. Our method is inspired by the Classic IoT control system and data drive recommendation systems.

We began by building the device that measured weather conditions. We connected the temperature and humidity sensor to a single board computer, Raspberry pi. The Raspberry pi sends the weather data to the Firebase database, which connects to the app we are coding called Smart Irrigation. The app then displays the real-time temperature and humidity. Using Python on the Jupiter Notebook, we utilized data that recorded the conditions in 36 cities in CA over five years to train a model with a machine learning algorithm that predicts the temperature and humidity of the upcoming week. When the app receives the temperature and humidity from the Firebase, the algorithm suggests whether or not the sprinkler system should be turned off, and from the app, the sprinkler system can be shut off.

There are some good features of our smart irrigation system [10]. First, we collected data from 2013 - 2020 in California to train the model, which is suitable to predict future California weather conditions and generate recommendations. Second, the Raspberry Pi provides stable control to the power of sprinkles which can reduce the cost and improve the production. Third, the mobile app is easy to access and operate, expanding the production influence. Therefore, we believe that the smart system we built can effectively solve the problem and help save water in California.

To demonstrate how the irrigation system works we used simulation software to show when the sprinklers start compared to the weather and the result of a lawn. The goal of the simulation was to show how using the sprinkler system you save water and money. By comparing the efficiency and cost effectiveness of the irrigation system to using normal sprinklers or manually watering. The efficiency can be measured by comparing the water wasted and time spent to the result of how the lawn looks. After comparing the irrigation systems, the results are almost the same but using our irrigation system was far superior. The time spent and the water wasted manually watering a lawn was way higher than the automatic systems by a large margin. Both our system

and other smart irrigation systems had much greater efficiency with lower water wastage, but ours was slightly better. Other smart irrigation systems check the humidity of the soil and past weather patterns, but run on a timer so it isn't saving as much water as possible. Our system checks the humidity and past weather patterns, but only turns on during the morning when watering is most effective and on days when watering is necessary without the lawn degrading. Because of this, it is slightly more effective at saving as much water as possible. Also, our system is far more cost efficient since instead of buying a hundred dollar control unit, one can just buy a few pieces of hardware and download an app.

The rest of the paper is organized as follows: Section 2 details the challenges that we met during the experiment and designing the sample; Section 3 focuses on the details of our solutions corresponding to the challenges that we outlined in Section 2; Section 4 presents the relevant details about the experiment we conducted, followed by the related works discussed in Section 5. Finally, Section 6 gives the concluding remarks, as well as discussing the future of this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. How to use Python and Raspberry Pi

One challenge that we faced was learning how to learn to use python and how to use the Raspberry Pi. Since I was unfamiliar with Python Programming, I had to learn how python worked and what the imported modules did. Figuring out which things to import and how to use them was difficult since I had never imported modules or used their functions. Another problem that we encountered was how to use the Raspberry Pi. Since we were new to using Raspberry Pi's everything from connecting it to the WiFi to connecting it to the Firebase was new. Everything we tried always somehow had small problems that we ran into that were very tedious to fix. For example, the IP address for some reason kept changing every time we restarted the Raspberry Pi, so we had to figure out how to keep it from changing. Connecting the circuit to the Raspberry Pi was also new, but I figured it out in the end.

2.2. How to connect the flutter app with the server

At first, we created a flutter app to run the program but it would not connect with the server. Because of this we had to change it to a thunkable app and a Firebase database. The thunkable interface was different and we faced many challenges when using the code, since many of the functions did not appear. The instructions put on the website did not appear to help the problem, but after a while we eventually figured out how to use the functions connecting the code to the firebase. Creating the program that could take values from the Firebase and display it on an app was also a challenge since we didn't know what most of the functions did. After researching what the things did, we were able to create a code to update the app from the readings in the Firebase and display it on the screen in real time.

2.3. How to find data and make it usable

Another challenge we faced was in finding data and making our data usable for the machine learning algorithm. First, most of the resources we found only included temperature data, but since we were looking for both temperature and humidity together, all of those resources were unusable. When we finally found data that included temperature and humidity, we found that the numbers produced results almost opposite of what we expected-- during the day the temperature

was cooler than during the night. This was definitely wrong, so we investigated and found that all the data was recorded with the same time zone, regardless of what time it actually was in the area being observed. After figuring out what time zone the data was recorded in, we needed to make all of the data fit the actual time in the area, the final challenge in making the data usable for the machine learning algorithm was in writing code so that all of the cities we were using would have the correct time correlation to the weather and humidity.

3. SOLUTION

SmartIrrigation is a smart Irrigation control system based on an IoT system and driven by big data [14]. The settings of the irrigation system can be controlled through the mobile app. The customers can get suggestions from the cloud based on the big data computing result to drive the on/off setting of the system.

SmartIrrigation integrates with the database, mobile app, and machine learning cloud computing server.

The main control system uses a raspberry pi embedded device with a temperature sensor and a humidity sensor to detect weather conditions. The data will be sent to the Firebase database and will sync up to the mobile app that was developed with flutter with a machine learning algorithm. To train the algorithm, we analyzed data from 36 cities from California for the past 7 years. We analyzed everything from the most general overview down to the details to provide accurate suggestions that apply to California cities.

Therefore, it can be used for detecting the weather conditions, predicting the temperature and humidity of certain days based on data, and performing analysis tasks such as whether to control the sprinklers by person or by algorithm to save water.

Additional relevant tasks are to manage all the systems besides one city. The main technical challenge of the system is managing the uploading and storing data —while delivering the real-time tem and hum data necessary for cloud computing. Furthermore, the accurate rate of the machine learning algorithm must be more than 95% to make the prediction stable. To achieve these goals, our tool consists of three main components (see the thick boxes in Figure 1):

- a hierarchical data structure for storing the tem and hum attributes in an aggregated format [15];
- an stable cloud server with trained data to improve the efficient
- a powerful mechanism for efficiently turning on and off the controller.

We also provide a set of navigation techniques for exploring the graph. The following sections describe these components in detail.

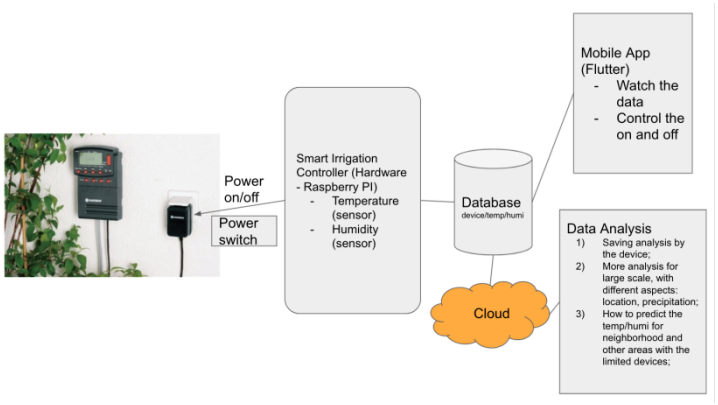

Figure 1. Overview of the system

We created a Flutter app to monitor and receive the data from the sensor, here are 3 screens we have:

1. The first screen is the login screen, which is used to login to find the data from your sensor.

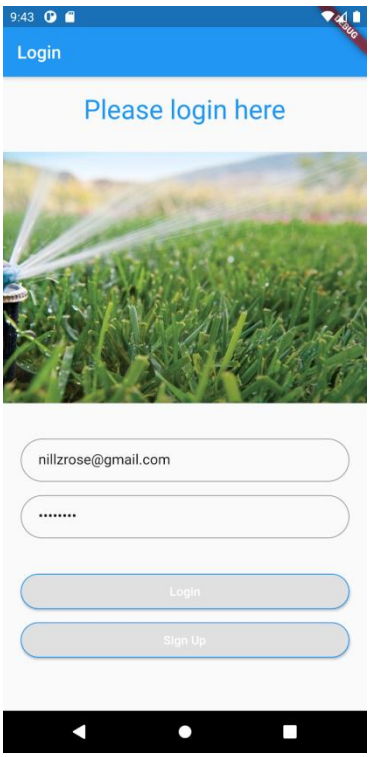

Figure 2. Screenshot of login page

```
margin: EdgeInsets.only(top:10, left:20, right: 20),
child: RaisedButton(
  shape: RoundedRectangleBorder(
    borderRadius: BorderRadius.circular(30),
    side: BorderSide(color: Colors.blue)), // RoundedRectangleBorder
  textColor: Colors.white,
  child: Text("Login"),
  onPressed: () {
    print("Email:");
    print(emailController.text);
    print("Password:");
    print(passwordController.text);
    FirebaseAuth.instance.signInWithEmailAndPassword(
      email: emailController.text.trim(), password: passwordController.text).th
        print("Login Successful");
        print(val.toString());
        print("userid is");
        print(val.user.uid);
        Navigator.push(context, MaterialPageRoute(builder: (context) => MyHomeP
      }).catchError((error) {
        print("error");
        print(error.toString());
      });
```

Figure 3. Screenshot of code (1)

2. The second one is the Homepage Screen, which loads the data….. and will update the data…. The 3 buttons function is listed below:

1) Refresh Button: Refresh the data from pi and sensor
2) ON/OFF: Control the Pi by app
3) Get Suggestion: Get Suggestion from Server to turn on or off

Figure 4. Screenshot of Home page

```
FlatButton(
  child: Text("on/off"),
  color: Colors.blueAccent,
  textColor: Colors.white,
  onPressed: () {
    on_status = !on_status;
    print("after flip");
    print(on_status);
    FirebaseDatabase.instance.reference().update(
      {
        'on' : on_status,
      }).then((val) {
        print("Data Set Successfully");
        setState(() {});
      }).catchError((error) {
        print(error);
      });
  },
), // FlatButton
SizedBox(height: 10),
FlatButton(
  child: Text("Get suggestion"),
  color: Colors.blueGrey,
  textColor: Colors.white,
  onPressed: () {
    Navigator.push(context, MaterialPageRoute(builder: (context) => suggestionPage()));
  },
), // FlatButton
```

Figure 5. Screenshot of code 2

3. Suggestion Page, you can input the suggestion bar and see the recent rain and temperature here:

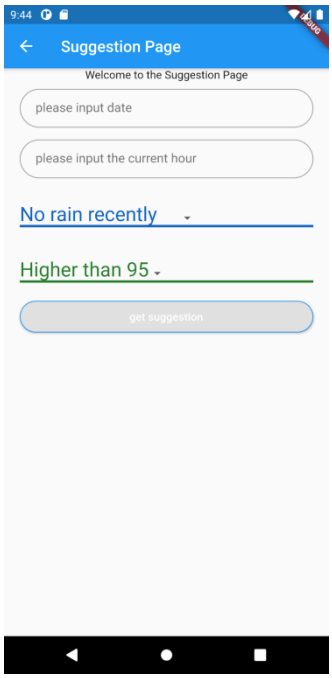

Figure 6. Screenshot of Suggestion page

```
Container(
  width: 400,
  height: 40,
  margin: EdgeInsets.only(top:10, left:20, right: 20),
  child: DropdownButton<String>(
    value: temRange,
    icon: Icon(Icons.arrow_drop_down),
    iconSize: 40,
    elevation: 16,
    style: TextStyle(color: Colors.green),
    underline: Container(
      height: 20,
      color: Colors.green
    ), // Container
    onChanged: (String newValue) {
      setState(
        () {
          temRange = newValue;
        });
    },
    items: <String>["Higher than 95", "85-95", "70-85", "50-70", "Lower than 50"]
      .map<DropdownMenuItem<String>>((String value) {
        return DropdownMenuItem<String>(value: value, child: Column(
          children: [
            SizedBox(height:5),
            Text(value)
          ],
        )); // Column // DropdownMenuItem
        //Sizebox(height:5), Text(value));
      }
    ).toList()
  ) // DropdownButton
), // Container
```

Figure 7. Screenshot of code 3

4. EXPERIMENT

To give a better suggestion, we use the python algorithm and put it on a suggestion response server. We analyze the data from 2013 - 2016 and give the suggestion based on what we learned on the curve.

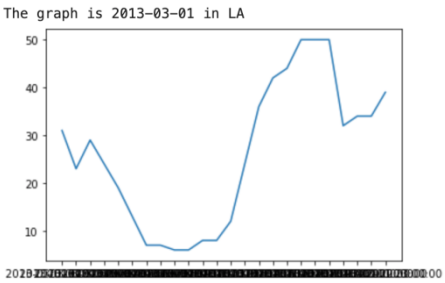

Figure 8. A Hum changes in one day of LA

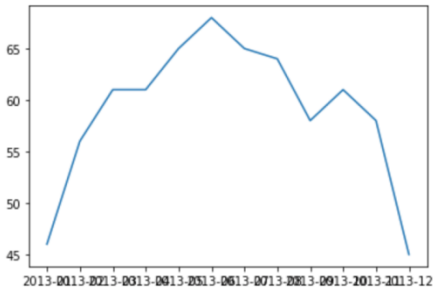

Figure 9. A Average Hum changes in one Year of LA

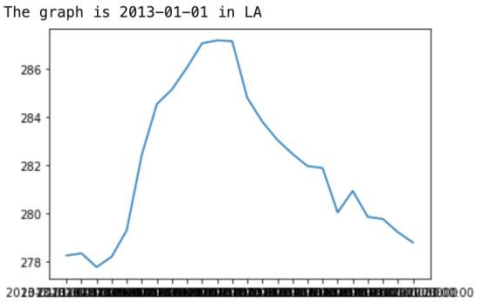

Figure 10. Temperature changes in one day of LA

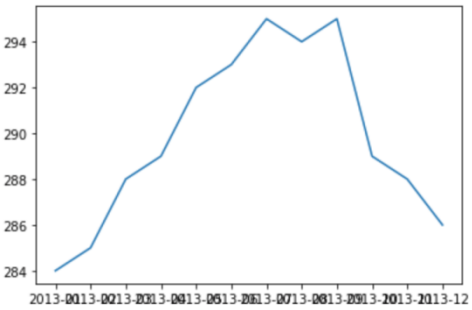

Figure 11. An Average Temperature changes in one Year of LA

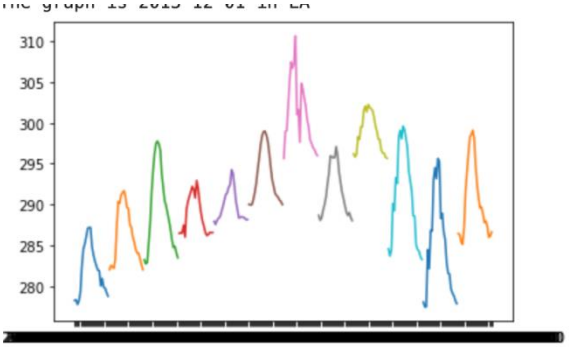

Figure 12. An Average Potential temperature changes by months

Based on these graphs we can determine when it is best to use water to conserve as much water as possible and save more money. The data we do the analysis comes from the website of the US Statistics. Based on the above analysis, we have selected a total of 5 different methods for irrigation recommendations. To suggest by A Hum changes in one day of LA, Average Hum changes in one Year of LA, Average Temp changes in one Year of LA, Average Hum changes in one Year of LA, Potential tem changes by months. Result shows that suggestion based on both Average Temp changes in one Year of LA and Average Hum changes in a day of LA have the highest prediction accuracy. So our algorithm used both Average Temp changes in one Year and Average Hum changes in a day as the data to train the suggestion model.

5. RELATED WORK

On the Smart Irrigation System, written by S. Darshna, T.Sangavi, Sheena Mohan, A.Soundharya, Sukanya Desikan, they create a system that monitors the amount of soil moisture and temperature using a predefined range of soil moisture and temperature that can be varied with soil or crop type [10]. The watering system can be turned on or off in case the moisture or temperature of the soil deviates from the desired range. When the soil is dry or has a high temperature, it will activate the irrigation system to pump water and bring the soil back into the desired range.

Kriti Taneja; Sanmeet Bhatia created an automated irrigation system using Arduino sensors to efficiently utilize water during irrigation [11]. The system has a soil moisture sensor for the soil near the plants and a water level sensor in a water container where water will be pumped to the plants. They created an algorithm using maximum values of soil moisture to control the quantity of water in the soil and the water level sensor to measure the amount of water being sent from the tank. Using an Arduino with an ATMega328 micro controller, they can use automatic irrigation to turn the pumping motor in the tank on or off depending on the dampness of the soil. By doing so, it eliminates human intervention and saves more water by efficiently and effectively irrigating the plants. The micro controller collects values from the soil sensors and depending on the values, it pumps water out. A LCD screen is connected to the micro controller to display the values from the soil and water pump. The water level sensor is used so that the water tank always contains enough water to efficiently irrigate the crops.

In a paper written by KK Namala, Krishna Kanth Prabhu A V, Anushree Math, Ashwini Kumari, Supraja Kulkarni, they propose a smart irrigation system that can be used to control the watering of plants, so their humans can be less human intervention [12]. They focus on the wastage of water and saving as much as possible, which is a problem in modern times. It also helps save time, is cost effective, protects the environment, and is low maintenance with a low operating cost which results in an efficient irrigation service. The Raspberry Pi is used to make the system compact and sustainable. It uses a sensor to measure the moisture of soil and based on the desired moisture it can switch a relay that controls a solenoid valve for irrigation.

6. CONCLUSIONS

We have created a temperature and humidity app that detects factors such as temperature and humidity to decide if it is a good or bad time to use water [12]. This app can be used to conserve as much water as possible during the California drought. By saving water it not only helps out the world's water problem and it saves as much money on water bills as possible.

One variable that affects the accuracy is that the temperature data is limited to past values in the Los Angeles area. This means that the data is only accurate in Los Angeles and other areas for the most part, inaccurate. This makes it unusable in other areas across the country and the globe. Also, the current device is pretty impractical. It must be connected to a power source and have a WiFi connection. This isn’t practical as the device has to be outside to measure temperature and humidity values. Also, using raspberry pi is quite expensive and along with all the other components the cost adds up.

To solve the limitations many things can be done. The historical data can be expanded to many major cities across America or the World, and in time it will be able to accurately function in all cities [13]. Another solution to the practicality is by using an Arduino instead of a Raspberry Pi. Since the device only performs one function, an Arduino would be better suited as it is only able to run one program. This could also bring the cost down.

REFERENCES

[1] Mishra, Ashok K., and Vijay P. Singh. "A review of drought concepts." Journal of hydrology 391.1-2 (2010): 202-216.

[2] Blanke, Amelia, et al. "Water saving technology and saving water in China." Agricultural water management 87.2 (2007): 139-150.

[3] Darshna, S., et al. "Smart irrigation system." IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) 10.3 (2015): 32-36.

[4] Joorabchi, Mona Erfani, Ali Mesbah, and Philippe Kruchten. "Real challenges in mobile app development." 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. IEEE, 2013.

[5] Barrett, Christopher B., Michael R. Carter, and C. Peter Timmer. "A century-long perspective on agricultural development." American Journal of Agricultural Economics 92.2 (2010): 447-468.

[6] Van Lente, Harro. "Promising technology: The dynamics of expectations in technological developments." (1995): 0741-0741.

[7] Shamir, Uri Y., and Charles DD Howard. "Water distribution systems analysis." Journal of the Hydraulics Division 94.1 (1968): 219-234.

[8] Thompson, Allen L., et al. "Testing of a water loss distribution model for moving sprinkler systems." Transactions of the ASAE 40.1 (1997): 81-88.

[9] Romero, Cristobal, and Sebastian Ventura. "Data mining in education." Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 3.1 (2013): 12-27.

[10] Darshna, S., et al. "Smart irrigation system." IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) 10.3 (2015): 32-36.

[11] Taneja, Kriti, and Sanmeet Bhatia. "Automatic irrigation system using Arduino UNO." 2017 International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE, 2017.

[12] Yamazoe, Noboru, and Yasuhiro Shimizu. "Humidity sensors: principles and applications." Sensors and Actuators 10.3-4 (1986): 379-398.

[13] Simonton, Dean Keith. "Qualitative and quantitative analyses of historical data." Annual review of psychology 54.1 (2003): 617-640.

[14] Buhl, Hans Ulrich, et al. "Big data." Business & Information Systems Engineering 5.2 (2013): 65-69.

[15] Tanimoto, Steven, and Theo Pavlidis. "A hierarchical data structure for picture processing." Computer graphics and image processing 4.2 (1975): 104-119.