1 Derivations for a Map $\phi : \{0,1\}^{32} \rightarrow R^D$ Mapping 32-bit 0/1-Valued Challenge Vectors to D-Dimensional Feature Vectors

Given the challenge for an arbiter PUF, we need to map it to a new feature vector that gives a linear relation with the upper signal delay $t_u(c)$.

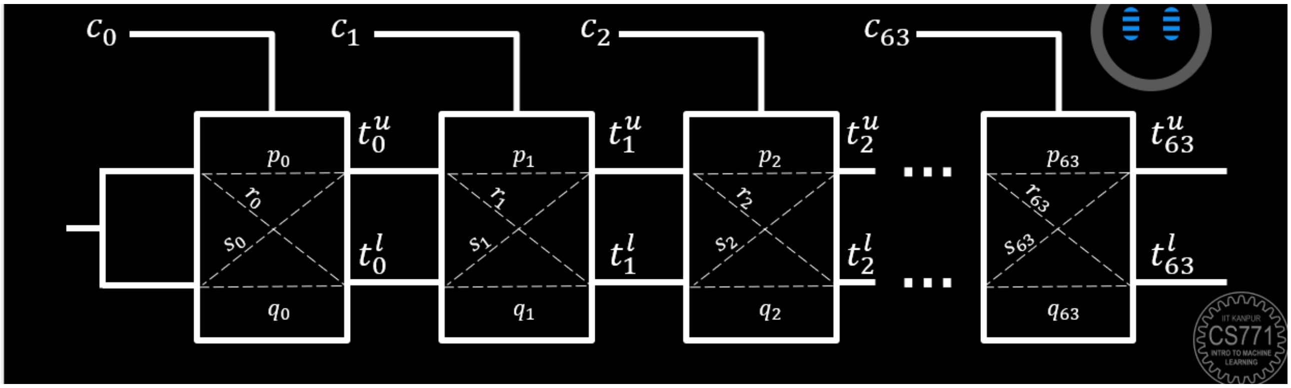

Figure 1: respective delay constants per puff

Delay Definitions

For a given stage i, let:

- let τ_i be the total delay at stage i
- t_u^i be the delay for the upper signal at stage i
- t_l^i be the delay for the lower signal at stage i

The delays can be expressed as:

$$\tau_i = t_u^i + t_l^i$$

Time Delay Equation

Given a challenge vector $c \in \{0,1\}^{32}$, we define:

$$d_i = 1 - 2c_i$$

The time delay τ_i for the upper signal at stage i can be expressed as:

$$\tau_i = (1 - c_i)(p_i + q_i) + c_i(s_i + r_i) + \tau_{i-1}$$

This can be rewritten using d_i:

$$\tau_i = (1 - \frac{1-d_i}{2})\alpha_i + \left(\frac{1-d_i}{2}\right)\beta_i + \tau_{i-1}$$

where:

$$\alpha_i = p_i + q_i$$

$$\beta_i = s_i + r_i$$

This simplifies to:

$$\tau_i = \gamma_i + d_i\delta_i + \tau_{i-1}$$

where:

$$\gamma_i = (\frac{\alpha_i + \beta_i}{2})$$

$$\delta_i = \frac{\alpha_i - \beta_i}{2}$$

$$\tau_1 = \gamma_1 + d_1\delta_1 + \tau_0$$

$$\tau_0 = \gamma_0 + d_0\delta_0$$

Linear Model Representation

To derive the linear model, we start making a patter for example :

$$\tau_1 = C + d_1\delta_1 + d_0\delta_0$$

where:

$$C = \gamma_1 + \gamma_0$$

similarly for any i <= 32:

$$\tau_i = \sum_{j=0}^{i}(\delta_j d_j + \gamma_j)$$

$$\tau_{32} = \sum_{i=1}^{32}(\delta_i d_i + \gamma_i)$$

Let:

$$\mathbf{W}' = (\delta_1, \delta_2, \ldots, \delta_{32})$$
$$\mathbf{x}' = (d_1, d_2, \ldots, d_{32})$$
$$b' = \sum_{i=1}^{32}\gamma_i$$

Then:

$$\tau_{32} = \mathbf{W'}^T\mathbf{x}' + b'$$

Now, Δ_i is the difference between upper delay and lower delay at the ith puff i.e.,:

$$\Delta_i = t_i^u - t_i^l$$

where:

$$t_i^u = (1-c)(t_{i-1}^u + p_i) + c(t_{i-1}^l + r_i)$$
$$t_i^l = (1-c)(t_{i-1}^l + q_i) + c(t_{i-1}^l + r_i)$$

the expression for Δ_i is:

$$\Delta_i = d_i * \Delta_{i-1} + a_i * d_i + b_i$$

where:

$$a_i = \frac{(p_i - q_i + r_i - s_i)}{2}$$
$$b_i = \frac{(p_i - q_i - r_i + s_i)}{2}$$

Generalizing:

$$\Delta_{32} = \sum_{i=1}^{32} W_i * x_i + b_{63}$$
$$= \mathbf{W}^T\mathbf{x} + b$$

where:

$$x_i = d_i * d_{i+1} * d_{i+2} \ldots . d_{32}$$

The absolute delay for the upper signal can be expressed as:

$$t_u^{32} = \frac{\tau_{32} + \Delta_{32}}{2} = \frac{\mathbf{W'}^T\mathbf{x}' + \mathbf{W}^T\mathbf{x} + b' + b}{2}$$

$$= \frac{\mathbf{W'}^T\mathbf{x}' + \mathbf{W}^T\mathbf{x} + b_0}{2}$$

Similarly, for the lower signal:

$$t_l^{32} = \frac{\tau_{32} - \Delta_{32}}{2} = \frac{\mathbf{W'}^T\mathbf{x}' - \mathbf{W}^T\mathbf{x} + b' - b}{2}$$
$$= \frac{\mathbf{W'}^T\mathbf{x}' - \mathbf{W}^T\mathbf{x} + b_0'}{2}$$

Thus, we have derived the linear model for predicting the delay of the upper signal in an arbiter PUF.

2 Dimensionality of question 1

the dimensionality of $\mathrm{t}_u^{32} willbe := 32 + 32 - 1$

$$= 63$$

The term (-1) is due to one overlapping feature i.e. d_{32} which is a part of both the features $\mathbf{x}'$ and $\mathbf{x}$.

Apart from this we have a constant bias term

3 Predicting response0 and response1 using linear model from question 1

Response for both lower signals (Response0)
Response0 is 0 when signal from PUF0 (lower puff) reaches first and 1 when signal from PUF1 reaches first. i.e.,

$$Res0 = \frac{1 + \text{sign}(\omega^T x + b)}{2} \tag{1}$$

where $(\omega^T x + b) < 0$ if signal from PUF0 reaches first, i.e.,

$$(t_l^0)^{32} < (t_l^1)^{32}$$

which means time taken by upper signal of PUF1 is greater than that of PUF0.

$$\omega^T x + b = (t_l^0)^{32} - (t_l^1)^{32}$$

$$\omega^T x + b = \frac{\tau_{32}^0 - \Delta_{32}}{2}^0 - \frac{\tau_{32}^1 - \Delta_{32}^1}{2} + b$$

$$\omega^T x + b = \frac{(\Delta_{32}^1 - \Delta_{32}^0) + (\tau_{32}^0 - \tau_{32}^1)}{2} + b$$

$$= \frac{(\omega'^{0T} - \omega'^{1T})\mathbf{x}' - (\omega^{0T} - \omega^{1T})\mathbf{x}}{2} + b$$

Since, x is same for upper & lower PUF therefore, $\Phi(c)$ is also same
similarly, for Response1 we can comapare the upper signals from PUF0 and PUF1:
if the upper signal of PUF0 reach first response1 will be 0 otherwise 1

$$\omega^T x + b = (t_u^0)^{32} - (t_u^1)^{32}$$

$$\omega^T x + b = \frac{\tau_{32}^0 + \Delta_{32}}{2}^0 - \frac{\tau_{32}^1 + \Delta_{32}^1}{2} + b$$

$$\omega^T x + b = \frac{(\Delta_{32}^0 - \Delta_{32}^1) + (\tau_{32}^0 - \tau_{32}^1)}{2} + b$$

$$= \frac{(\omega'^{0T} - \omega'^{1T})\mathbf{x}' + (\omega^{0T} - \omega^{1T})\mathbf{x}}{2} + b$$

therefore the Response0 and Response1 can be expressed using time taken by upper PUF signals and lower PUF signals.

4 Dimensionality of Response0 and Response1

This was same as that of abosolute time i.e., 63 for both the responses. Since we only had to subtract two model outputs with same feature, thus the dimensionality did not change.

5 Solution for Part 5:

Zipped Solution to Assignment 1

6 Comparison of LinearSVC and Logistic Regression Hyperparameter Effects

6.1 Comparison Based on Loss Function

The tables below summarize the average training time and test accuracy for different loss functions and regularization values for both LinearSVC and Logistic Regression.

Logistic Regression :

Loss Function	C Value	Avg. Train Time (s)	Avg. Test Accuracy
hinge	low	0.1604	0.9880
hinge	medium	0.1496	0.9918
hinge	high	0.1603	0.9966
squared hinge	low	0.1438	0.9885
squared hinge	medium	0.1786	0.9875
squared hinge	high	0.1730	0.9936

Table 1: Average Training Time and Test Accuracy for Logistic Regression

Logistic Regression :

Loss Function	C Value	Avg. Train Time (s)	Avg. Test Accuracy
hinge	low	0.4210	0.9860
hinge	medium	0.9144	0.9940
hinge	high	1.7721	0.9888
squared hinge	low	0.4018	0.9853
squared hinge	medium	0.4758	0.9880
squared hinge	high	0.4744	0.9883

Table 2: Average Training Time and Test Accuracy for LinearSVC

Analysis and Possible Reasons for Differences From the results, we observe the following:

- **Logistic Regression:** The hinge loss function with a high `C` value yielded the highest test accuracy of 0.9966, with an average training time of 0.1603 seconds. The squared hinge loss function also performed well but slightly lower in accuracy and varied more in training time.
- **LinearSVC:** The hinge loss function with a medium `C` value resulted in the highest test accuracy of 0.9940, but the training time was significantly higher (0.9144 seconds) compared to Logistic Regression. The squared hinge loss function showed consistent performance but was generally lower in accuracy compared to the hinge loss.
- **Possible Reasons:**
  - The hinge loss function is generally more robust and might provide better margin maximization, leading to higher accuracy.
  - The squared hinge loss might lead to more regularization but can sometimes underperform due to over-penalizing errors.

6.2 Comparison Based on Regularization Parameter (C)

The following tables provide the comparison of average training time and test accuracy based on different values of the regularization parameter C.

Logistic Regression :

C Value	Loss Function	Avg. Train Time (s)	Avg. Test Accuracy
low	hinge	0.1604	0.9880
medium	hinge	0.1496	0.9918
high	hinge	0.1603	0.9966
low	squared hinge	0.1438	0.9885
medium	squared hinge	0.1786	0.9875
high	squared hinge	0.1730	0.9936

Table 3: Comparison of Logistic Regression with Different C Values

LinearSVC :

C Value	Loss Function	Avg. Train Time (s)	Avg. Test Accuracy
low	hinge	0.4210	0.9860
medium	hinge	0.9144	0.9940
high	hinge	1.7721	0.9888
low	squared hinge	0.4018	0.9853
medium	squared hinge	0.4758	0.9880
high	squared hinge	0.4744	0.9883

Table 4: Comparison of LinearSVC with Different C Values

Analysis and Possible Reasons for Differences Observations from the results based on different C values:

- **Logistic Regression:** Increasing the C value generally improves test accuracy, with the highest accuracy observed at high C values. However, training times are relatively consistent across different C values.
- **LinearSVC:** Training times significantly increase with higher C values, especially for the hinge loss function. Test accuracy peaks at medium C values for hinge loss, while squared hinge loss shows more consistent but slightly lower accuracy.
- **Possible Reasons:**
  - Higher C values reduce the regularization effect, allowing the model to fit the training data more closely, leading to better accuracy but potentially higher risk of overfitting and longer training times.
  - Medium C values provide a balance between regularization and model complexity, often leading to optimal performance.

6.3 Conclusion

To sum up, both LinearSVC and Logistic Regression have their own strengths and weaknesses depending on the hyperparameters used. Generally, the hinge loss function tends to give better accuracy for both models. Logistic Regression shows consistent training times across different C values, whereas LinearSVC's training time varies significantly. Medium C values usually offer a good trade-off between regularization and accuracy. These results underscore the importance of careful hyperparameter tuning to get the best performance from a model.