אילן הינדי

[הקלד את תקציר המסמך כאן. התקציר הוא בדרך כלל סיכום קצר של תוכן המסמך. הקלד את תקציר המסמך כאן. התקציר הוא בדרך כלל סיכום קצר של תוכן המסמך.]

אלגוריתמים מבוזרים

הוספת אלגוריתם

הוספת אלגוריתם

[1 יצירת הקוד 3](#_Toc473026157)

[2 מחיקת אלגוריתם 3](#_Toc473026158)

[3 הקונספט של אחזקת נתונים על האלגוריתם 4](#_Toc473026159)

[4 תכנון האלגוריתם 5](#_Toc473026160)

[4.1 קביעת קונפיגורצית הרשת 5](#_Toc473026161)

[4.2 החלטה על סוגי ההודעות ומבנה ההודעות 6](#_Toc473026162)

[4.3 מעבד ((Processor 6](#_Toc473026163)

[4.3.1 החלטה אילו תכונות יש למעבד (תכונות שאינן משתנות במהלך הריצה) 7](#_Toc473026164)

[4.3.2 החלטה אילו משתנים משמשים בזמן ריצה 7](#_Toc473026165)

[4.4 ערוץ (channel) 8](#_Toc473026166)

[4.5 רשת (Network) 8](#_Toc473026167)

[5 תכנות האתחול 8](#_Toc473026168)

[6 פונקציות utility לתכנות אלגוריתם 8](#_Toc473026169)

[7 יצירת Documentation 9](#_Toc473026170)

[8 החלטה אילו פרמטרים נחוצים 9](#_Toc473026171)

[9 לבחור אילו הודעות יהיו במערכת 9](#_Toc473026172)

[10 ב – Process להכין פונקציות שבונות את ההודעות 9](#_Toc473026173)

[11 ב – Process לייצר פונקציה לאיתחול האלגוריתם 10](#_Toc473026174)

[12 ב – Process לייצר פונקציה לטיפול בקבלת הודעות 10](#_Toc473026175)

[13 פונקציות לשליחת הודעות 11](#_Toc473026176)

הוספת אלגוריתם

# יצירת הקוד

1. להכנס לתוכנית
2. בתפריט הראשי Program -> Add Algorithm
3. לצאת מהתוכנית
4. הקבצים נוצרו בכתובת

<your path>

\Distributed Algorithms\Program\DistributedAlgorithms

\DistributedAlgorithms\Algorithms Data\

<the subject name>\<The algorithm name>

1. להכליל את הקבצים החדשים ב – Visual Studio

# מחיקת אלגוריתם

מחיקת אלגוריתם יכולה להיעשות רק מתוך ה – Visual studio ע"מ לא לבלבל את ניהול הקבצים שלו

1. מחק את ה – directory של האלגוריתם / נושא מתוך ה – directory : Algorithms
2. מחק את ה – directory של האלגוריתם / נושא מתוך ה – directory : Algorithms Data
3. הפעל את התוכנית ומהתפריט הראשי לך ל – Program -> create config file from existing

# הקונספט של אחזקת נתונים על האלגוריתם

1. כל המחלקות המייצגות האלמנטים של הרשת יורשים מ – NetworkElement
2. ב – NetworkElement יש 5 מילונים מסוג AttributeDictionary של תכונות
   1. ElementAttributes – תכונות המוגדרות ע"י ה – base class (כלומר תכונות המשותפות לכל האלמנטים מאותו סוג
   2. PrivateAttributes – תכונות המוגדרות עבור אלגוריתם ספציפי
   3. OperationResults – תכונות המשתנות בזמן ריצה (כלומר משתנים של הריצה)
   4. OperationResultsBackup – גיבוי עבור ה – OperationResults ע"מ לאפשר הפעלה מחדש עם אותו אתחול של הפרמטרים
   5. PresentationParameters – תכונות של התצוגה. תכונות אילו משמשות לשינוי התצוגה בזמן ריצה
   6. PresentationParametersBackup – גיבוי ל – PresentationParameters (בזמן ריצה משתנים ה – PresentationParameters ה – PresentationParametersBackup משמש להחזרת התצוגה למצב של עריכה)
3. כל מילון מורכב ממפתח מסוג dynamic ומתכונה מסוג Attribute
4. הגדרת המפתחות
   1. המפתח צריך להיות Enum ה – Enums מוגדרים באופן הבא:
   2. שם ה – Enum עבור מילון <dictionary name> יוגדר מפתח <dictionary name>Keys
5. הגדרת המילונים
6. במחלקות (מלבד מחלקות היורשות מ – BaseMessage) מוגדרים המילונים שניתן להוסיף להם ערכים

public enum PrivateAttributeKeys { }

public new enum OperationResultKeys { }

1. עבור ההודעות (מחלקות היורשות מ – BaseMessage) מוגדרים המילונים הבאים:

public new enum MessageTypes {}

public new enum FieldTypes { }

1. סוגים אפשריים לתכונה
   1. סוגים פשוטים
   2. Enum
   3. אלמנט של רשת (יורש מ – NetworkElement)
   4. רשימה של Attribute
   5. AttributeDictionary
2. גישה לערך הסבר הדוגמא:
   1. האובייקט של NetworkElement – message
   2. המילון ElementsAttributes
   3. המחלקה בה מוגדר ה – enum של המפתחות BaseMessage
   4. שם ה – enum של המפתחות ElementAttributeKeys
   5. שם המפתח Round

message.ElementAttributes[BaseMessage.ElementAttributeKeys.Round].Value

# תכנון האלגוריתם

## קביעת קונפיגורצית הרשת

לרשת יש 2 מאפיינים

* האם היא Centralized
* האם היא Directed

למרות שהפרמטרים המתארים את סוג הרשת הם תכונות לכל דבר מומלץ לקבוע אותם בתוכנה באמצעות שינוי הפרמטרים בפונקציה

protected override void CreateInitNetwork()

{

base.CreateInitNetwork();

// Determin network type parametes

ElementAttributes[BaseNetwork.ElementAttributeKeys.DirectedNetwork].Value = false;

ElementAttributes[BaseNetwork.ElementAttributeKeys.Centrilized].Value = true;

// The network has to have at lease one initiator

Processes[0].ElementAttributes[BaseProcess.ElementAttributeKeys.Initiator].Value = true;

}

## החלטה על סוגי ההודעות ומבנה ההודעות

1. לכל סוג של הודעה יש להוסיף כניסה ב – enum : MessageTypes של מחלקת ההודעות
2. לכל סוג של שדה בהודעות יש להוסיף כניסה ב – enum : FieldTypes

## מעבד ((Processor

### החלטה אילו תכונות יש למעבד (תכונות שאינן משתנות במהלך הריצה)

לכל תכונה יש לבצע :

1. הוספת התכונה ב – enum : PrivateAttributeKeys
2. הוספת אתחול לתכונה במתודה init לפי הדוגמא:

public override void Init(int idx, bool clearPresentation = true)

{

base.Init(idx, clearPresentation);

// Add initialization of PrivateAttributes here

PrivateAttributes.Add(PrivateAttributeKeys.NumberOfRounds, new Attribute { Value = 1});

}

### החלטה אילו משתנים משמשים בזמן ריצה

לכל משתנה יש לבצע :

1. הוספת המשתנה ב - enum: OperationResults
2. הוספת המשתנה במתודה init לפי הדוגמא:

protected override void InitOperationResults()

{

base.InitOperationResults();

// Add initialization of OperationResults here

OperationResults.Add(OperationResultKeys.Round, new Attribute { Value = 0, Editable = false});

OperationResults.Add(OperationResultKeys.Status, new Attribute { Value = Status.SentForwaredMessages});

}

## ערוץ (channel)

תוספת תכונות ומשתנים בדומה למעבד

## רשת (Network)

תוספת תכונות ומשתנים בדומה למעבד

# תכנות האתחול

1. ע"מ שתהליך יהיה יוזם התכונה של BaseProcess.ElementAttributeKeys.Initiator צריכה להיות true. את התכונה ניתן לעדכן בזמן בניית הרשת
2. בפונקציה של ה – Process בשם RunAlgorithm יש לתכנת מה ה – initiator מבצע ע"מ לאתחל את ריצת האלגוריתם (אין צורך לבדוק האם התהליך הוא initiator)

# תכנות הטיפול בהודעות

1. בעקרון לאחר האתחול האלגוריתם הוא לולאה שמבצעת איטרציה בכל פעם שמתקבלת הודעה ואין breakpoint.
2. בכל פעם שמתבצעת איטרציה נקראת הפונקציה RecieveHandling
3. האלגוריתם של הפונקציה הזאת הוא בד"כ (אך לא מחויב)
4. אם קבלת הודעה מסוג... בצע ...
5. אם קבלת הודעה מסוג ... בצע ...
6. וכו'

# פונקציות utility לתכנות אלגוריתם

# יצירת Documentation

# החלטה אילו פרמטרים נחוצים

1. פרמטרים שקיימים כבר ורוצים לשנות את ערכם בפונקציה init
2. פרמטרים שרוצים להוסיף:
   1. להוסיף ל – enum של PrivateAttributes
   2. לאתחל בפונקציה init
3. משתנים שרוצים להוסיף
   1. להוסיף ל – enum של OperationResults
   2. לאתחל בפונקציה InitOperationResults

# לבחור אילו הודעות יהיו במערכת

1. לכל message להוסיף כניסה ב- enum MessageType
2. אם רוצים שדות נוספים למה שיש להוסיף כניסה ל - MessageFieldKeys

# ב – Process להכין פונקציות שבונות את ההודעות

1. איתחול הודעה באמצעות:

|  |
| --- |
| אתחול הודעה מהודעה בשינוי הערוץ השולח |
| public BaseMessage(BaseMessage sourceMessage, BaseChannel sendingChannel):base(sourceMessage) |
| אתחול הודעה סטנדרטית ע"י נתינת כל השדות |
| public BaseMessage(dynamic messageTypeKey,  dynamic messageType,  int sourceProcess,  int sourcePort,  int destProcess,  int destPort,  int round,  int logicalClock) |
| אתחול הודעה עם אותם הפרמטרים בשינוי הערוץ של השליחה מס' הסבב והשעון הלוגי |
| public BaseMessage(dynamic messageTypeKey, dynamic messageType, BaseChannel channel, int round, int logicalClock) |

1. תוספת שדות להודעה המאותחלת באמצעות
2. public void AddField(dynamic key, Attribute attribute)
3. שינוי שדות באמצעות שינוי רגיל של התכונות של ההודעה

# ב – Process לייצר פונקציה לאיתחול האלגוריתם

1. הפונקציה תופעל על תהליכים שמסומנים initiators בלבד (אין צורך לבדור האם תהליך הוא initiator בפונקציה)

# ב – Process לייצר פונקציה לטיפול בקבלת הודעות

# פונקציות לשליחת הודעות

|  |
| --- |
| שליחת הודעה מוכנה |
| protected void Send(int destProcessId, BaseMessage message) |
| שליחת הודעה מוכנה לכל השכנים (הפרמטרים שמתארים את המקור והיעד ישתנו) |
| protected void SendToNeighbours(BaseMessage message, List<int> exclude) |
| בניית הודעה ושליחה לכל השכנים |
| protected void SendToNeighbours(dynamic messageType, AttributeDictionary fields, List<int> exclude, int round, int logicalClock) |