บทที่ 3
ซี ยู ทราฟฟิค ซิมูเลชั่น โปรแกรม

โดยทั่วไปโปรแกรมทางด้านซิมูเลชั่น เป็นโปรแกรมที่มีการทำงานที่จำลองจากสภาพจริง โปรแกรมจะทำงานตามเงื่อนไขที่กำหนดให้ในลักษณะซ้ำๆกัน การทำงานของซิมูเลชั่นทางด้านการจราจร ก็มีลักษณะการทำงานเหมือนซิมูเลชั่นทั่วไปก็คือ ทำงานจำลองสภาพการจราจรบนถนนโดยจำลองยวดยานเคลื่อนที่ซ้ำๆกันไป ลักษณะการวนซ้ำก็คือ ดำเนินการจัดการซ้ำทุกๆช่วงเวลา ดังนั้นการจำลองการจราจรโดยใช้ซิมูเลชั่นก็คือ จำลองสภาพการจราจรที่เกิดขึ้นในคาบเวลาหนึ่งๆที่ต้องการศึกษา ผลลัพธ์ที่ได้คือ ค่าพารามิเตอร์ที่บ่งบอกสภาพการจราจรที่ต้องการ อันเนื่องมาจากสภาพการจราจรที่กำหนดให้เป็นข้อมูลเริ่มต้นในกระบวนการทำซิมูเลชั่น

การทำซิมูเลชั่นทางด้านการจราจรสามารถจำแนกแบบจำลอง ตามแนวทางที่ใช้ในการจำลองสภาพการจราจรได้ 2 ประเภท คือ

- Microscopic Model เป็นแบบจำลองที่มีแนวทางในการจำลองสภาพการจราจรโดยเน้นในรายละเอียดเกี่ยวกับพฤติกรรมของยวดยานแต่ละคัน และเน้นพฤติกรรมระหว่างยวดยานแต่ละคันที่มีต่อกัน
- Macroscopic Model เป็นแบบจำลองที่มีแนวทางในการจำลองสภาพการจราจรที่เน้นพฤติกรรมยวดยานที่มีลักษณะเป็นกลุ่ม ไม่ละเอียดลงไปถึงพฤติกรรมการเคลื่อนตัวยวดยานแต่ละคันดังเช่น แบบจำลองประเภทแรก

จากแบบจำลองทั้ง 2 ประเภท ที่กล่าวมาแล้วนั้น การนำแบบจำลองไปใช้จึงมีความเหมาะสมและขีดจำกัดแตกต่างกัน แบบจำลองประเภท Microscopic Model นั้นเหมาะสมกับการจำลองสภาพการจราจรที่ต้องจำลองพฤติกรรมของยวดยานแต่ละคันๆไป แต่การที่นำไปทำเป็นโปรแกรมเพื่อจำลองการจราจรด้วยคอมพิวเตอร์นั้น จำเป็นต้องพิจารณาถึงเวลาที่ใช้ในการทำซิมูเลชั่นว่านานเกินไปหรือไม่ และหน่วยความจำของเครื่องคอมพิวเตอร์ว่ามีเพียงพอหรือไม่ เมื่อเปรียบเทียบระหว่างแบบจำลองทั้งสองแบบในกรณีที่จำลองโครงข่ายถนนเดียวกันและคาบเวลาที่จำลองเท่ากัน แบบจำลองประเภท Microscopic ต้องใช้เนื้อที่หน่วยความจำและเวลาในการทำซิมูเลชั่นมากกว่าแบบจำลองประเภท Macroscopic แบบ

จำลองประเภท Microscopic นั้นนิยมใช้ในการจำลองพฤติกรรมการจราจรที่เน้นในรายละเอียด เช่น การจำลองพฤติกรรมการจราจรบริเวณทางแยกเดียว การจำลองการจราจรบริเวณทางขึ้น-ลงทางด่วน เป็นต้น ส่วนแบบจำลองประเภท Macroscopic นั้น นิยมนำไปใช้ในการจำลองพฤติกรรมการจราจรที่ต่อเนื่องกันเป็นโครงข่าย ทั้งนี้เนื่องจากผลกระทบของพฤติกรรมยวดยานแต่ละคันมิได้ส่งผลโดยตรงต่อการจราจรในระดับที่เป็นโครงข่าย อีกทั้งสามารถลดจำนวนเนื้อที่หน่วยความจำและเวลาที่ต้องใช้ในการทำงานโปรแกรมลงไปได้มาก ทำให้สามารถทำเป็นโปรแกรมเพื่อใช้งานกับเครื่องคอมพิวเตอร์ ระดับไมโครคอมพิวเตอร์ได้โดยไม่เกิดปัญหาเรื่องหน่วยความจำไม่เพียงพอ

โปรแกรม ซี ยู ทราฟฟิค ซิมูเลชั่น เป็นโปรแกรมที่จัดอยู่ในประเภท Macroscopic Model ใช้จำลองสภาพการจราจรที่ต่อเนื่องกันเป็นโครงข่าย ข้อมูลที่ใช้ในการทำงานของโปรแกรมมีลักษณะคล้ายกับโปรแกรมซิมูเลชั่นการจราจรบริเวณทางแยกคือ

- ข้อมูลทางด้านกายภาพของโครงข่ายถนน คือ ความต่อเนื่องของโครงข่าย ความยาวของโครงข่าย เวลาที่ใช้เดินทาง (Free Flow หรือ Observed Speed) บนช่วงถนน ความจุของถนน จำนวนช่องทาง ความหนาแน่นสูงสุด (Jam Density)
- ข้อมูลทางด้านสัญญาณไฟแต่ละทางแยก คือ รอบเวลา (Cycle Time) จังหวะสัญญาณไฟ (Phasing Time) ค่าออฟเซทของสัญญาณไฟ (Offset Time) ทิศทางการเคลื่อนตัวของยวดยาน
- ข้อมูลปริมาณการไหลของยวดยาน คือ จำนวนยวดยานที่เข้าสู่โครงข่ายแต่ละช่วงเวลา สัดส่วนของยวดยานที่เลี้ยวบริเวณทางแยก
- เวลาที่ใช้ในการทำงานของโปรแกรม และคาบเวลาที่ส่งผลลัพธ์จากการทำงานของโปรแกรม

เหตุผลที่สำคัญในการเลือกใช้ ซี ยู ทราฟฟิค ซิมูเลชั่น โปรแกรม เป็นเครื่องมือในการศึกษาคือ เพื่อศึกษาการทำงานของโปรแกรมที่เป็นการการสะสมความรู้จากการวิจัยอย่างต่อเนื่องกันมาของหน่วยวิจัยจราจรและขนส่ง เพื่อให้สามารถจำลองการติดตั้งอุปกรณ์ตรวจวัดสภาพการจราจรหรือ "detector" บนถนนได้ ทำให้สามารถนำผลลัพธ์ดังกล่าวไปใช้ในการวิเคราะห์หาความสัมพันธ์กับความยาวคิว

โปรแกรม ซี ยู ทราฟฟิค ซิมูเลชั่น เป็นโปรแกรมที่มีขนาดใหญ่ จึงต้องแบ่งการทำงานออกเป็นส่วนๆตามหน้าที่ อยู่ในรูปการทำงานของโปรแกรมย่อย ผังการทำงานระหว่าง

โปรแกรมย่อยดังกล่าว แสดงในรูปที่ 3.1

การทำงานของโปรแกรมย่อยที่สลับซับซ้อนมาก สามารถแทนด้วยโปรแกรมย่อยอีกชั้นหนึ่งก็ได้ เพื่อทำให้เข้าใจง่ายในการพัฒนาโปรแกรมในอนาคต

3.1 การทำงานของโปรแกรมหลัก

จากรูปที่ 3.1 โปรแกรมเริ่มการทำงานโดยการอ่านแฟ้มข้อมูล การตั้งชื่อแฟ้มข้อมูลเข้า-ออกของโปรแกรม โดยผู้ใช้โปรแกรมเป็นผู้กำหนด

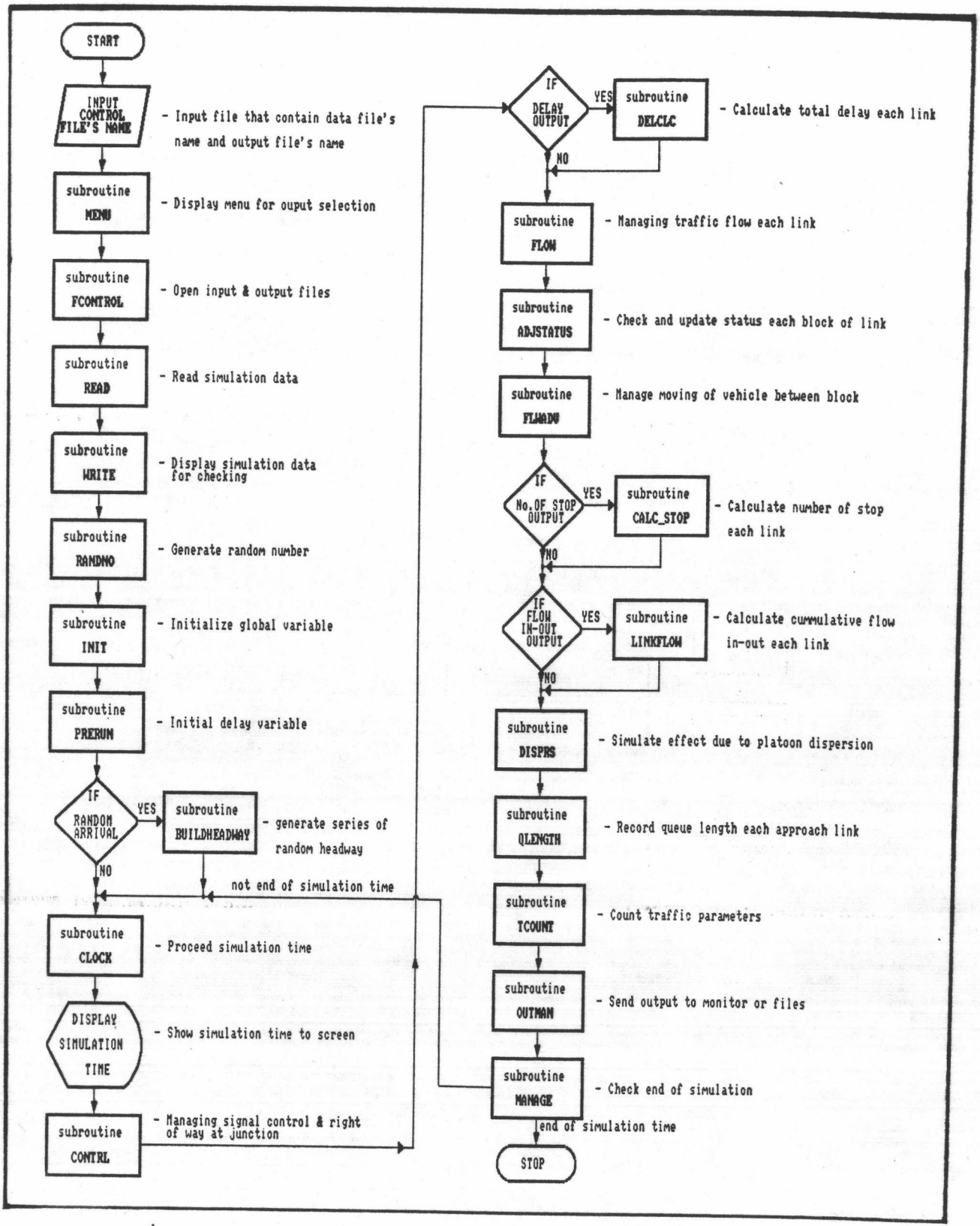
ต่อมา โปรแกรมจะอนุญาตให้ผู้ใช้ทำการเลือก ผลลัพธ์ที่ต้องการจากการทำชิมูเลชั่น โดยเก็บผลลัพธ์ไว้ในแฟ้มข้อมูลที่ตั้งชื่อไว้ในขั้นแรก โปรแกรมย่อยที่ทำหน้าที่นี้คือ โปรแกรมย่อย MENU หลังจากนั้นจึงส่งผลไปที่โปรแกรมย่อย FCONTROL ให้จัดการเปิดแฟ้มผลลัพธ์ เตรียมไว้

โปรแกรมย่อย READ ทำหน้าที่อ่านข้อมูลที่ใช้ในการทำงานของโปรแกรม แล้วแสดงผลออกมาให้ผู้ใช้ตรวจสอบความถูกต้องโดยโปรแกรมย่อย WRITE ซึ่งในขั้นตอนนี้ถ้าข้อมูลที่อ่านเข้ามาถูกต้องเรียบร้อยก็จะดำเนินการในขั้นต่อไป ผังการทำงานของโปรแกรมย่อย READ และ WRITE แสดงในรูปที่ 3.2 และ 3.3 ตามลำดับ

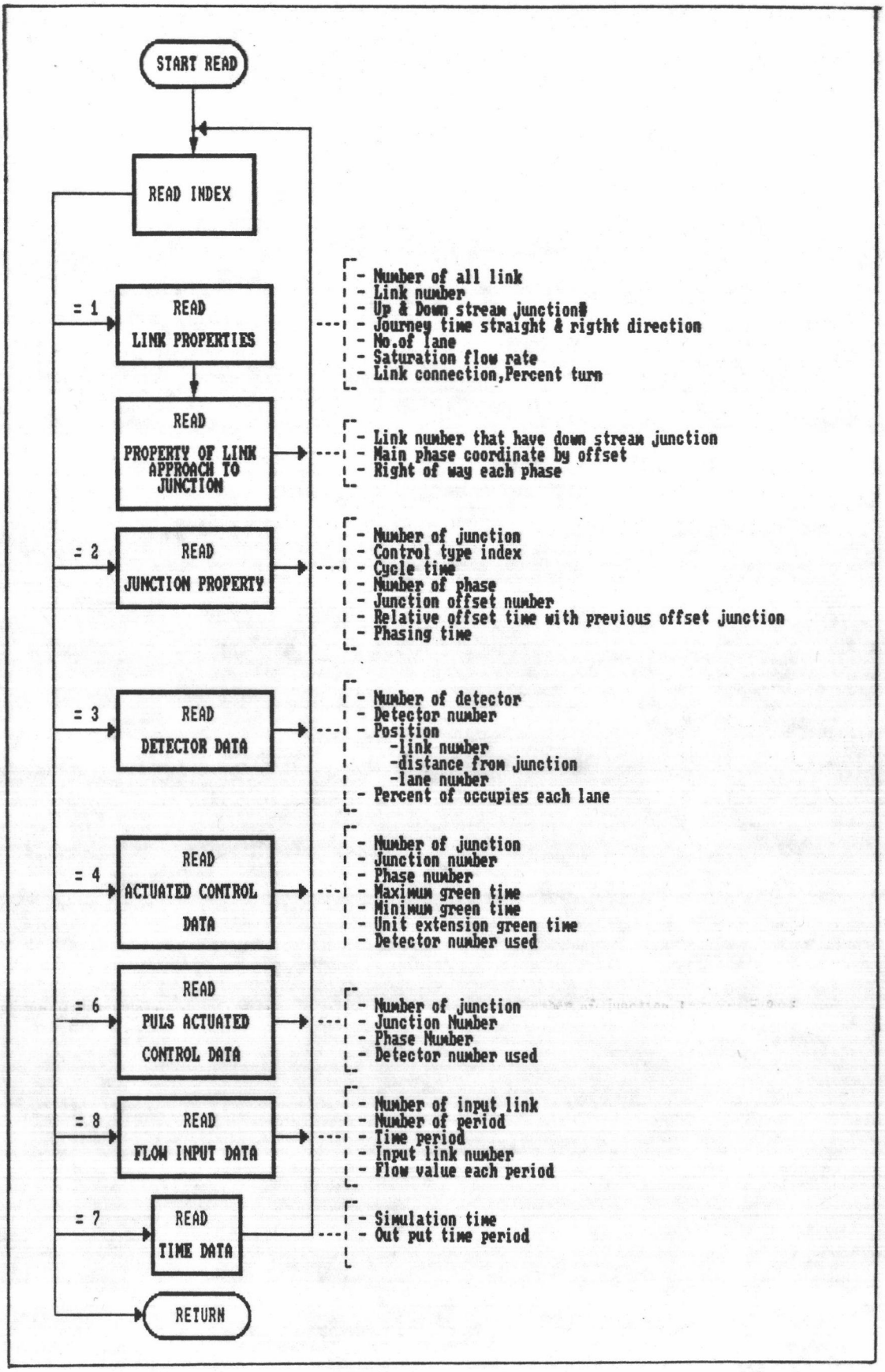
โปรแกรมย่อยในกลุ่มถัดมาทำหน้าที่จัดเตรียมค่าเริ่มต้นของตัวแปรที่ใช้ในการทำงาน คือ โปรแกรมย่อย RANDNO ทำหน้าที่สร้างตัวเลขสุ่ม (Random Number) ขึ้น 1,000 ชุด ส่วนโปรแกรมย่อย INIT กับ PRERUN ทำหน้าที่กำหนดค่าเริ่มต้นให้กับตัวแปรทุกตัวที่ใช้ในโปรแกรมขั้นตอนนี้จำเป็นอย่างยิ่ง เนื่องจากถ้าขาดขั้นตอนนี้ไป โปรแกรมจะนำเอาค่าเดิมค้างอยู่ในหน่วยความจำมาใช้ทำให้การทำงานของโปรแกรมผิดพลาด ผังการทำงานของโปรแกรมย่อย INIT แสดงในรูปที่ 3.4

ในขั้นต่อมา ถ้าผู้ใช้กำหนดประเภทการเกิดยวดยานเข้าสู่โครงข่าย (Arrival type) แบบสุ่ม (Random) โปรแกรมหลักจะทำการเรียกใช้โปรแกรมย่อย BUILDHEADWAY เพื่อสร้างตารางการเกิดยวดยานแบบสุ่มไว้ใช้ในขั้นตอนการสร้างยวดยานเข้าสู่โครงข่าย (Vehicle Generation)

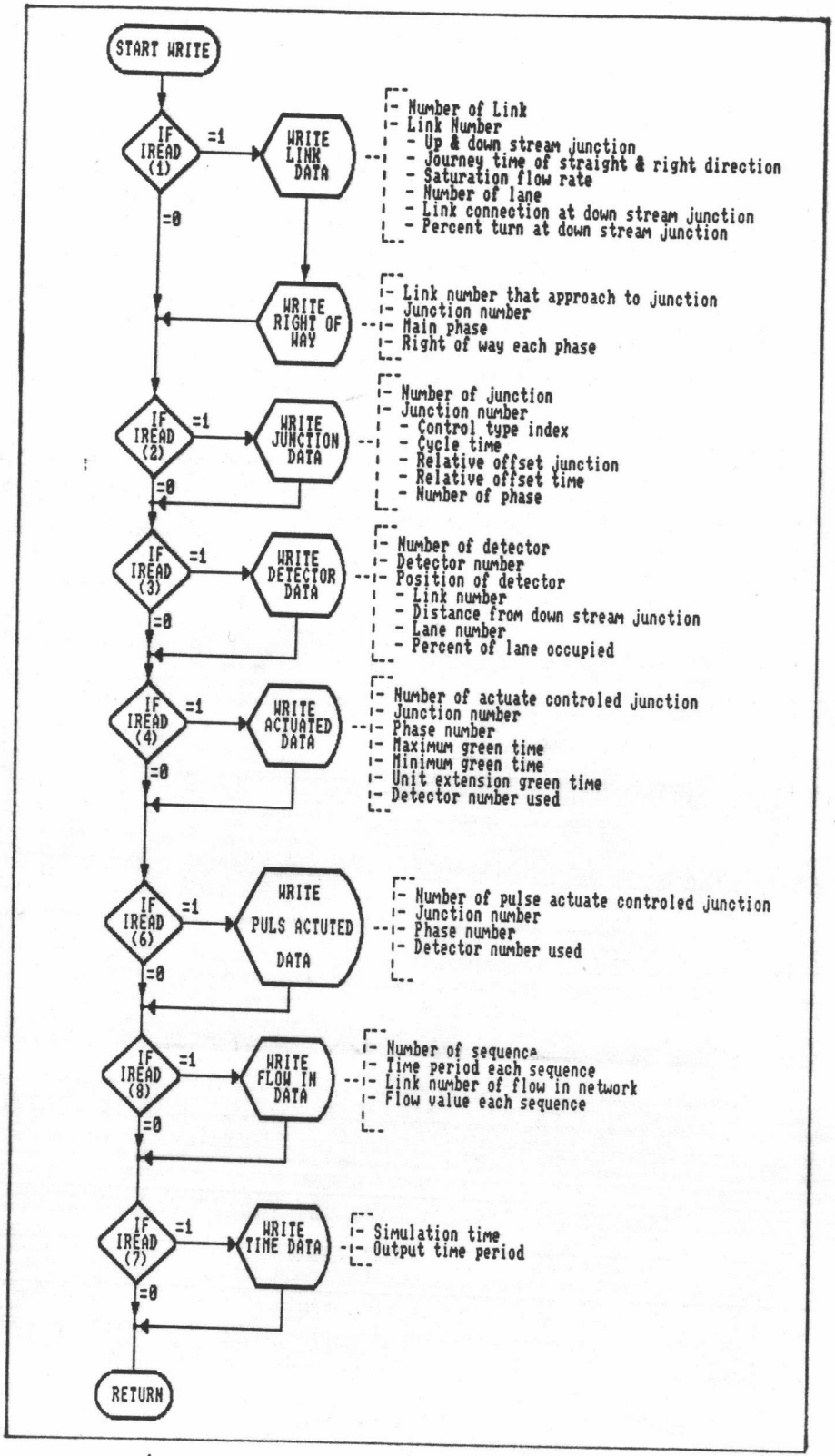
ในส่วนถัดมาเป็นส่วนของการทำงานหลักของ ซี ยู ทราฟฟิค ชิมูเลชั่น ซึ่งมีลักษณะทำงานที่วนซ้ำกลับมาเป็นรอบๆ จนกว่าจะครบกำหนดเวลาการทำชิมูเลชั่น โปรแกรมย่อยดัง



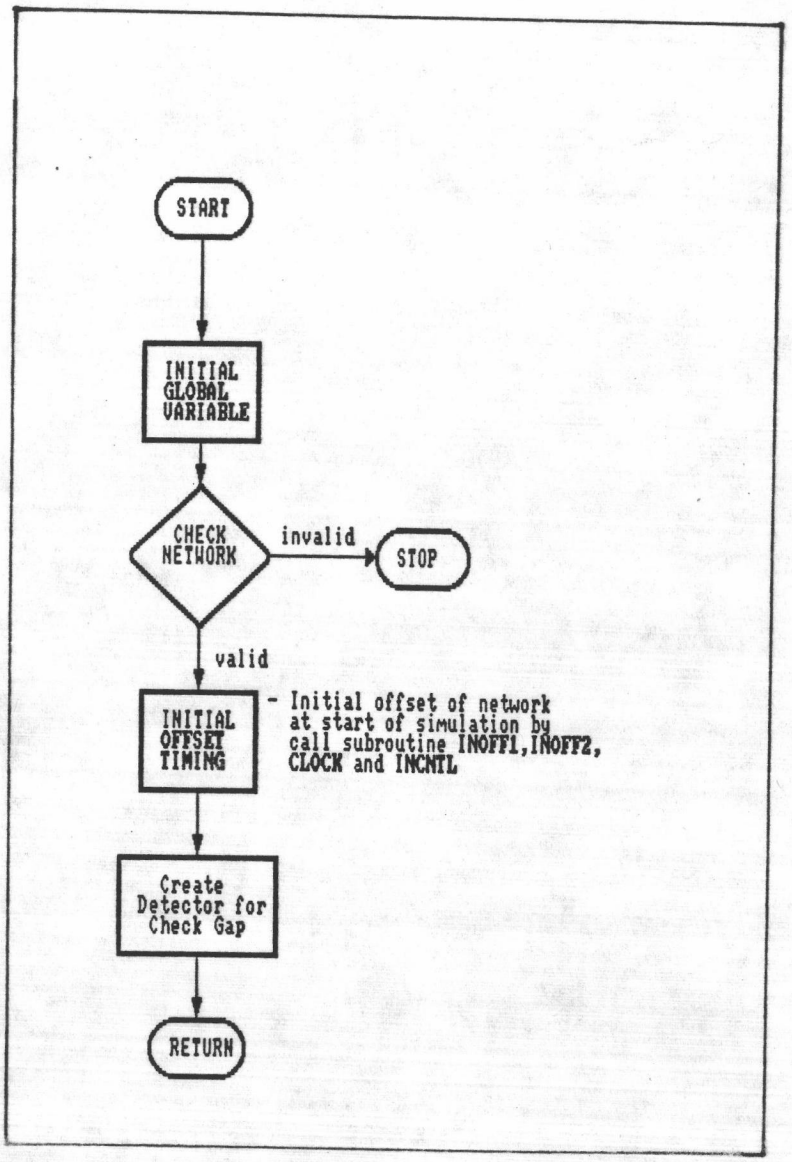
รูปที่ 3.1 ผังการทำงานภายในโปรแกรมหลักของ ซี ยู ทราฟฟิค ซิมูเลชั่น



รูปที่ 3.2 ผังการทำงานภายในโปรแกรมย่อย READ



รูปที่ 3.3 ผังการทำงานภายในโปรแกรมย่อย WRITE



รูปที่ 3.4 ผังการทำงานภายในโปรแกรมย่อย INIT

กล่าวเรียงลำดับดังต่อไปนี้คือ

- โปรแกรมย่อย CLOCK ทำหน้าที่เพิ่มเวลาขึ้น 1 หน่วยเวลา เมื่อการทำงานวนครบ 1 รอบ หน่วยเวลาดังกล่าวในงานวิทยานิพนธ์นี้ใช้ 1 วินาที แล้วแสดงผลออกทางจอภาพเพื่อรายงานสถานการทำงานของโปรแกรม

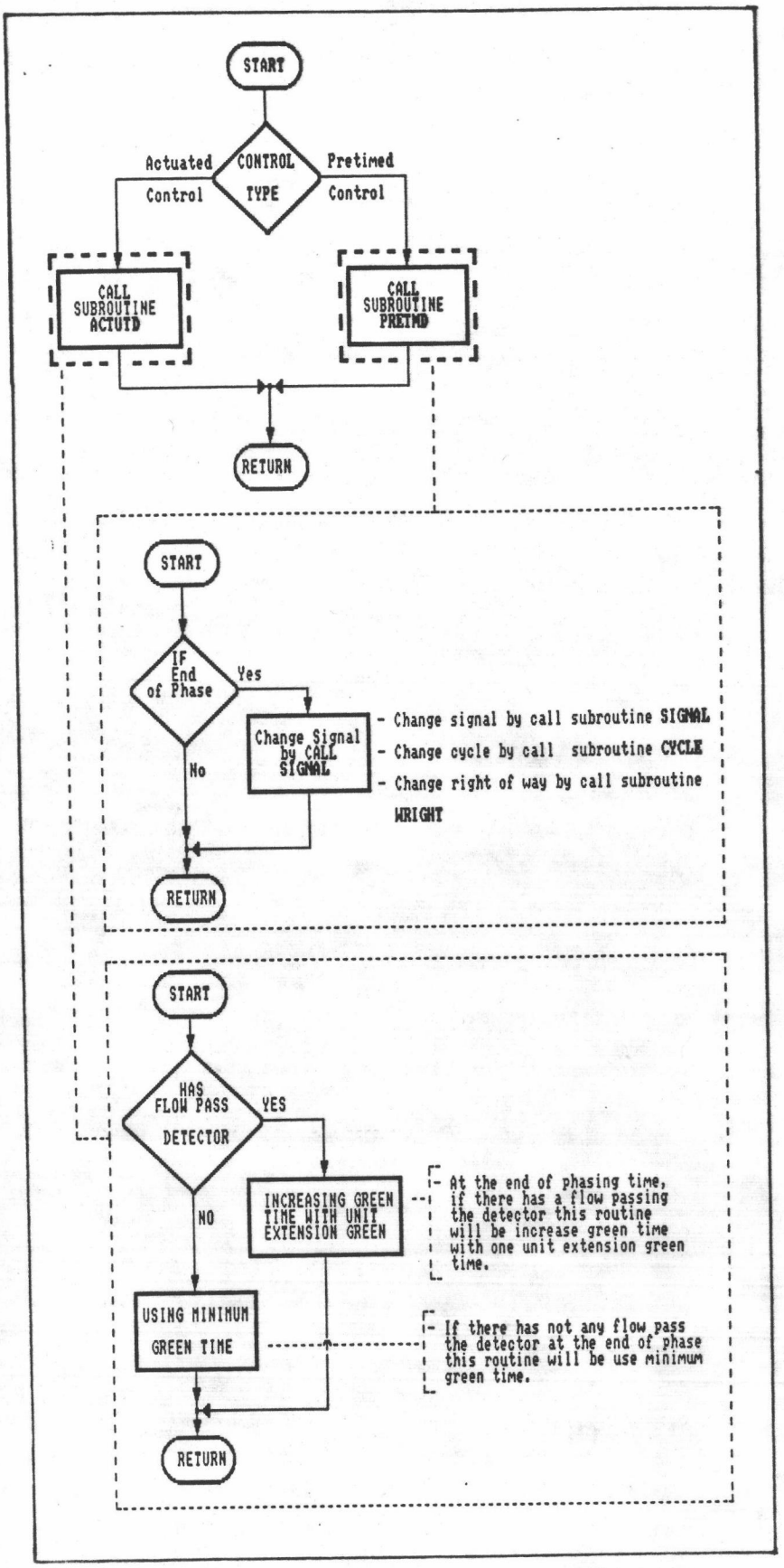
- โปรแกรมย่อย CONTRL ทำการจำลองระบบสัญญาณไฟที่ควบคุมทางแยก โดยเรียกใช้โปรแกรม เพื่อทำหน้าที่ควบคุมสัญญาณไฟบริเวณทางแยกตามประเภททางแยก ตามประเภท การควบคุมที่กำหนดจากข้อมูล คือ โปรแกรมย่อย PRETMD สำหรับสัญญาณไฟประเภท Fixed Time โปรแกรมย่อย ACTUATD สำหรับสัญญาณไฟประเภท Semi - Actuated Control ผังการทำงานแสดงในรูปที่ 3.5

- โปรแกรมย่อย DELCLC ทำหน้าที่รวบรวมค่าความล่าช้ารวม (Total Delay) ที่เกิดขึ้นทุก Link ที่เข้าสู่ทางแยก เป็นโปรแกรมย่อยที่ถูกเรียกใช้ในกรณีที่ผู้ใช้ต้องการทราบค่าความล่าช้า การคำนวณค่าความล่าช้าทำโดยการเปรียบเทียบการเคลื่อนตัว ผ่านทางแยกในกรณีที่ควบคุมด้วยสัญญาณไฟ กับกรณีที่ไม่มีการควบคุมด้วยสัญญาณไฟ

- โปรแกรมย่อย FLOW ดำเนินการเกี่ยวกับการเคลื่อนตัวของยวดยานแต่ละ Link เดิมโปรแกรมย่อย FLOW ยังทำหน้าที่เรียกโปรแกรมย่อยอื่นๆที่ดำเนินการเกี่ยวกับการเคลื่อนที่ของยวดยานอีกด้วย ในงานวิทยานิพนธ์ฉบับนี้ได้แยกโปรแกรมย่อยบางส่วนที่ถูกเรียกโดยโปรแกรมย่อย FLOW ออกมาขึ้นโดยตรงกับโปรแกรมหลัก เพื่อดึงหน้าที่การทำงานออกมาแสดงให้ชัดเจน ไม่ขึ้นต่อโปรแกรมย่อย FLOW จนทำให้โปรแกรมย่อย FLOW มีขนาดที่ใหญ่และสลับซับซ้อนจนเกินไป โปรแกรมย่อยที่ถูกเรียกใช้โดยโปรแกรมย่อย FLOW มีดังนี้ คือ โปรแกรมย่อย RTFLOW โปรแกรมย่อย INFLOW โปรแกรมย่อย OTFLOW โปรแกรมย่อย ADJSTATUS โปรแกรมย่อย FLWADV โปรแกรมย่อย CALC_STOP โปรแกรมย่อย LINKFLOW และ โปรแกรมย่อย DISPRS

- โปรแกรมย่อย RTFLOW ทำหน้าที่ดำเนินการเกี่ยวกับการเคลื่อนตัวของยวดยานในช่องทางเลี้ยวขวา ของทุก Link และจัดการกรณีที่ยวดยานหยุดในช่องทางเลี้ยวขวาจนเกินความยาวที่จัดไว้

- โปรแกรมย่อย INFLOW ดำเนินการจัดยวดยานเข้าสู่โครงข่าย และจัดเก็บปริมาณยวดยานที่ติดค้างอยู่นอกโครงข่ายไว้ โปรแกรมย่อยส่วนนี้ทำหน้าที่เรียกโปรแกรมย่อยที่



รูปที่ 3.5 ผังการทำงานภายในโปรแกรมย่อย CONTRL

สร้างยวดยาน (Vehicle Generation) ทั้งในกรณีของการเกิดยวดยานแบบสม่ำเสมอ (Uniform Arrival) และแบบสุ่ม (Random Arrival)

- โปรแกรมย่อย OTFLOW เป็นโปรแกรมย่อยที่มีขนาดใหญ่ และมีการทำงานค่อนข้างสลับซับซ้อน ดำเนินการควบคุมยวดยานที่เคลื่อนตัวออกจาก Link หนึ่งไปสู่ Link ถัดไปตามจังหวะสัญญาณไฟที่ได้รับ ควบคุมการเคลื่อนตัวของยวดยานทั้ง 3 ทิศทางคือ เลี้ยวซ้าย ตรง เลี้ยวขวา ในส่วนของยวดยานเลี้ยวขวามีส่วนที่ทำการจำลองการเลี้ยวขวาในขณะที่ยวดยานทิศทางที่สวนมา (Opposing Traffic) ได้รับสัญญาณไฟเขียว โดยอาศัยช่วงเวลาที่ห่างระหว่างยวดยานพอเพียง (Gap Acceptance) ผังการทำงานของโปรแกรมย่อย FLOW แสดงในรูปที่ 3.6

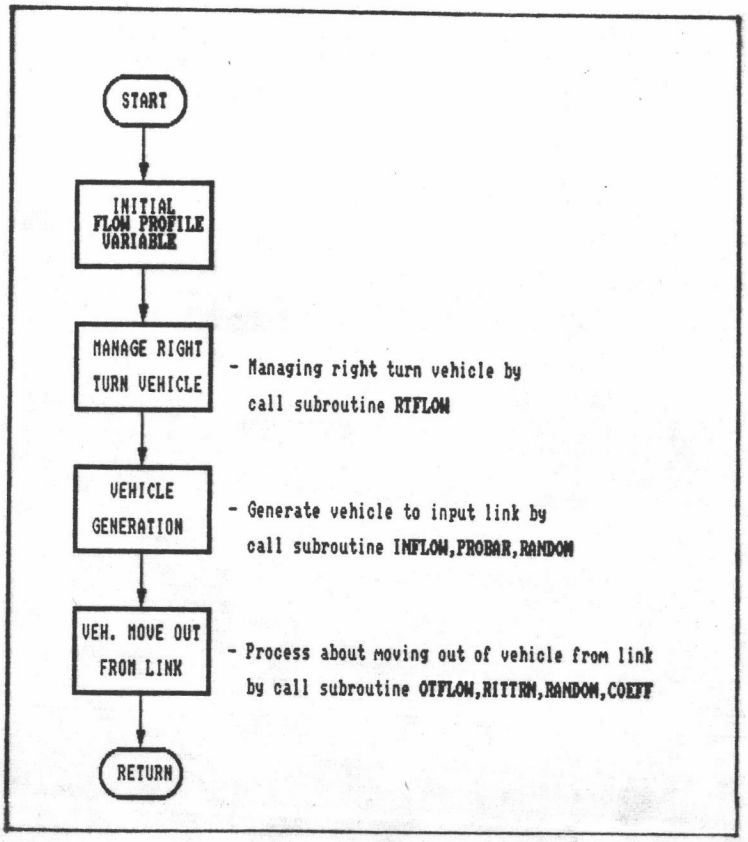
- โปรแกรมย่อย ADJSTAUS เป็นโปรแกรมย่อยที่ผู้ทำวิจัยได้แยกออกมาจากโปรแกรมย่อย FLOW เพื่อแสดงให้เห็นหน้าที่อย่างชัดเจน โดยทำหน้าที่ปรับสถานะภาพของช่อง (Block) แรกของทุก Link โดยตรวจสอบจากปริมาณยวดยานที่มีอยู่ใน Block เพื่อนำไปใช้พิจารณาในการเคลื่อนตัวของยวดยานระหว่าง Block ในโปรแกรมย่อย FLWADV

- โปรแกรมย่อย FLWADV เป็นโปรแกรมย่อยที่พิจารณาการเคลื่อนตัวของยวดยานระหว่าง Block เป็นคู่ๆ โดยเริ่มจาก Block ท้ายสุดของ Link มา Block ข้างหน้าสุด เป็นส่วนที่สำคัญของ ซี ยู ทราฟฟิค ซิมูเลชั่น หลักการทำงานที่เป็นข้อพิจารณาอยู่ในรายละเอียดหัวข้อถัดไป

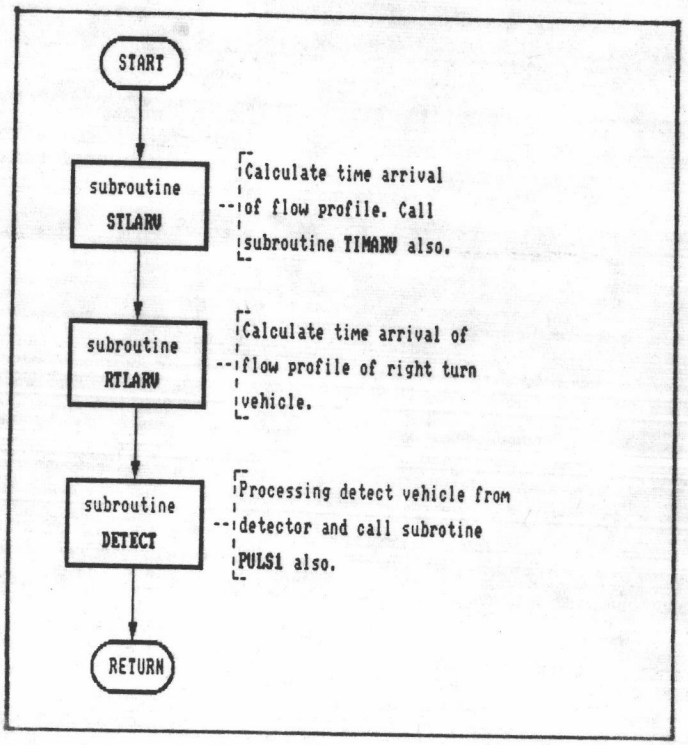
- โปรแกรมย่อย CALC_STOP เป็นโปรแกรมย่อยที่ทำหน้าที่คำนวณจำนวนครั้งที่ยวดยานหยุดในแต่ละ Link เรียกใช้เมื่อผู้ใช้โปรแกรมต้องการทราบผลของจำนวนครั้งที่ยวดยานหยุด

- โปรแกรมย่อย LINKFLOW ทำหน้าที่คำนวณปริมาณยวดยานสะสมที่เข้า-ออก Link แต่ละ Link เรียกใช้ในกรณีที่ผู้ใช้ต้องการทราบผลยวดยานเข้า-ออก Link

- โปรแกรมย่อย DISPRS เป็นโปรแกรมย่อยที่ทำหน้าที่คำนวณการกระจายตัวของยวดยานเมื่อเคลื่อนตัวไปบน Link หรือสภาพการณ์ที่เรียกว่า " Platoon Dispersion " ทำให้แบบจำลองการจราจรมีความสมบูรณ์ใกล้เคียงกับพฤติกรรมจริงๆของยวดยานบนท้องถนน ค่าคงที่ของการกระจายตัวกำหนดไว้ในโปรแกรมย่อย BLOCKDATA HO



รูปที่ 3.6 ผังการทำงานภายในโปรแกรมย่อย FLOW



รูปที่ 3.7 ผังการทำงานภายในโปรแกรมย่อย TCOUNT

- โปรแกรมย่อย QLENGTH ทำหน้าที่คำนวณความยาวของคิวที่เกิดขึ้นบน Link คำนวณจากสถานะของ Block ในแต่ละ Block ความยาวคิวเป็นความยาวคิวจากเส้นทางหยุด (Stop Line) ถึงตำแหน่ง Block ที่ยวดยานคันสุดท้ายเข้ามาจอด เป็นส่วนที่ผู้ทำวิจัยเพิ่มเติมขึ้นมาเพื่อใช้ประโยชน์ในการสร้างแบบจำลองการจัดการคิว

- โปรแกรมย่อย TCOUNT ทำหน้าที่บันทึกเวลาที่ยวดยานเข้าสู่ทางแยกเพื่อนำไปใช้ในการคำนวณค่าความล่าช้า โดยเรียกใช้โปรแกรมย่อย STLARV สำหรับรถช่องทางตรงและโปรแกรม RTLARV สำหรับยวดยานที่อยู่ในช่องทางเลี้ยวขวา ต่อจากนั้นทำหน้าที่เรียกใช้โปรแกรมย่อย DETECT เพื่อจำลองการทำงาน Traffic Detector แล้วคำนวณผลจากการตรวจวัดมาเก็บไว้ใช้ในการคำนวณช่วงเวลาที่ยวดยานเลี้ยวขวาต้องใช้เพื่อเลี้ยวตัดหน้ายวดยานที่วิ่งสวนมา (Gap Acceptance) ผังการทำงานแสดงในรูปที่ 3.7

- โปรแกรมย่อย OUTMAN ทำหน้าที่ส่งผลลัพธ์ที่ผู้ใช้ต้องการออกทางจอภาพและบันทึกลงแฟ้มข้อมูล โดยเลือกนำการแสดงผลลัพธ์เฉพาะที่ผู้ใช้ต้องการเท่านั้น โปรแกรมย่อย OUTMAN นี้ประกอบด้วยโปรแกรมย่อยเป็นกลุ่มๆ ตามผลลัพธ์ที่ต้องการ คือ

  . กลุ่มที่จัดการเกี่ยวกับค่าความล่าช้า ได้แก่ โปรแกรมย่อย VDEL และ OUTDEL โดย VDEL ทำหน้าที่คำนวณความล่าช้าประเภทต่างๆ ส่วน OUTDEL ส่งผลลัพธ์ออกทางจอภาพและแฟ้มข้อมูล

  . กลุ่มที่จัดการเกี่ยวกับ Detector ได้แก่ โปรแกรมย่อย DETPRM และ OUTDET โดย OUTDET คำนวณค่า Time Occupancy และค่าปริมาณจราจรที่ตรวจวัดได้จาก detector แต่ละตัวและส่งผลลัพธ์ออกโดย โปรแกรมย่อย OUTDET

  . กลุ่มที่จัดการเกี่ยวกับความยาวคิว ได้แก่ โปรแกรมย่อย OUTQUEUE โดยรวบรวมมาจากโปรแกรมย่อย QLENGTH แล้วส่งผลออกทางจอภาพและแฟ้มข้อมูล

  . กลุ่มที่จัดการเกี่ยวกับปริมาณยวดยานเข้า-ออก Link ได้แก่โปรแกรมย่อย OUTFLW ทำหน้าที่ส่งผลลัพธ์ที่รวบรวมได้จากโปรแกรมย่อย LINKFLOW ออกทางจอภาพ และแฟ้มข้อมูล

  . โปรแกรมย่อยกลุ่มที่จัดการเกี่ยวกับจำนวนครั้งที่หยุดของยวดยานแต่ละ Link

ได้แก่ โปรแกรมย่อย OUTSIP เป็นโปรแกรมย่อยที่ทำหน้าที่ส่งผลลัพธ์ออก โดยรวบรวมผลมาจากโปรแกรมย่อย CALC_STOP

- โปรแกรมย่อย MANAGE เป็นโปรแกรมย่อยที่พิจารณาว่าครบเวลาการทำงานของโปรแกรมหรือไม่ โดยตรวจสอบจากเวลาที่คำนวณไว้จากโปรแกรมย่อย CLOCK ถ้าครบแล้วก็ให้โปรแกรมหยุดทำงาน ถ้ายังไม่ครบให้โปรแกรมกลับไปเริ่มทำงานซ้ำที่โปรแกรมย่อย CLOCK จนกว่าจะครบ นอกจากนี้ยังทำหน้าที่เปลี่ยนค่าปริมาณการจราจรตามที่ผู้ใช้กำหนดไว้ให้เปลี่ยนแปลงไปตามช่วงเวลาและสร้างตารางการเกิดยวดยานขึ้นใหม่ กรณีที่ใช้ตารางเดิมจนครบ ซึ่งใช้กับการเกิดยวดยานแบบสุ่มเท่านั้น

3.2 หลักการทำงานของโปรแกรม

3.2.1 การจำลองโครงข่ายถนน และสัญญาณไฟ

เมื่อเริ่มต้นทำงานโปรแกรมจะทำการสร้างโครงข่ายถนน ซึ่งประกอบด้วย Node และ Link โดย Node แทนทางแยก ส่วน Link แทนถนน โดยแทนช่วงที่ต่อระหว่างทางแยก Link หนึ่งๆ หมายถึงถนนที่มีทิศทางการจราจรไปทางเดียวเท่านั้น ดังนั้นในกรณีช่วงถนนช่วงหนึ่งมีทิศทางการจราจรไป-กลับ 2 ด้าน จะแทนด้วย Link 2 เส้นเสมอ

คุณสมบัติต่างๆ ของ Link ที่ผู้ใช้โปรแกรมเป็นผู้กำหนดมีดังต่อไปนี้

- หมายเลขอ้างอิงของ Link กำหนดเป็นตัวเลขจำนวนเต็ม
- จำนวนช่องทาง ช่องทางปกติทั่วไปเป็นช่องทางที่มียวดยานในทิศทางตรง เลี้ยวซ้าย ส่วนกรณีที่มีการจัดช่องทางเลี้ยวขวาไว้เฉพาะนั้น โปรแกรมกำหนดให้มีได้เพียง 1 ช่องทางเท่านั้น
- ความยาวของ Link กำหนดเป็นเวลาที่ใช้เดินทางหน่วยเป็นวินาที คำนวณได้จากสูตร

$$\text{เวลาที่ใช้เดินทาง (วินาที)} = \frac{\text{ระยะทางของช่วงถนน (หน่วย เมตร)}}{\text{ความเร็วอิสระ (หน่วย เมตร/วินาที)}}$$

ความเร็วอิสระ (Free Flow Speed) เป็นค่าที่คำนวณจากความยาวของ Block กับช่วงเวลาที่เปลี่ยนแปลงเมื่อโปรแกรมทำงานครบ 1 รอบ (Scaning Time Interval)

ความเร็วอิสระ (เมตร/วินาที) = $\frac{\text{ความยาวของ Block (หน่วย เมตร)}}{\text{Scanning Time Interval (หน่วย วินาที)}}$

โปรแกรมใช้ค่าคงที่ดังนี้

- Scanning Time Interval = 1 วินาที
- ความยาวของ Block = 14 เมตร/Block

ดังนั้น ความเร็วอิสระ = 14/1 เมตร/วินาที
หรือเท่ากับ (14 x 3600 / 1000) = 50.4 กิโลเมตร/ชั่วโมง

ดังนั้นเวลาที่ใช้เดินทาง Link คำนวณจาก สูตร

เวลาในการเดินทางบน Link = $\frac{\text{ระยะทางของ Link (หน่วย เมตร)}}{14}$

ค่าที่คำนวณได้ปัดเศษให้เป็นจำนวนเต็ม แต่ใช้ได้ไม่เกิน 80 วินาที เนื่องจากข้อจำกัดในการจองหน่วยความจำของโปรแกรม ถ้าเวลาในการเดินทางเกิน 80 วินาที อาจจะทำการแก้ไขตัวโปรแกรม (Source Code) ให้สามารถจำลองกรณีที่เวลาในการเดินทางเกินกว่า 80 วินาที

- ความจุของ Link เป็นค่าที่กำหนดปริมาณสูงสุดที่ยวดยานสามารถเคลื่อนตัวจาก block หนึ่งไป block ถัดไป และใช้เป็นค่าที่กำหนดปริมาณยวดยานสูงสุดที่สามารถเคลื่อนตัวออกจาก Link ในหนึ่งหน่วยเวลา หรือที่เรียกว่า อัตราการไหลอิ่มตัว (Saturation Flow Rate) ค่าที่ใช้คำนวณเฉลี่ยจากอัตราการไหลอิ่มตัวของทุกๆ ช่องทางบริเวณทางแยก โดยเฉลี่ยรวมทั้งช่องทางตรง เลี้ยวซ้าย และ เลี้ยวขวา มีหน่วยเป็น คัน/ชั่วโมง/ช่องทาง ค่าที่นำไปใช้ต้องปัดทศนิยมให้เป็นเลขจำนวนเต็ม

- การเชื่อมต่อกับทางแยก โดยกำหนดหมายเลขทางแยก ทั้ง Upstream และ Downstream ในกรณีที่เป็น Input Link ซึ่งไม่มีทางแยก Upstream ซึ่งไม่มีทางแยก Upstream กำหนดให้ตัวเลขทางแยก Upstream เป็นศูนย์ ส่วนกรณี Output Link เช่นเดียวกันคือ กำหนดให้หมายเลขทางแยก Downstream ของ Link เป็นศูนย์

- การเชื่อมต่อกับ Link อื่นบริเวณทางแยก กำหนดเป็นหมายเลข Link ที่เข้ามาเชื่อมต่อทั้ง 3 ทิศทางเลี้ยวซ้าย ตรง และเลี้ยวขวา ในกรณีที่บางทิศทางไม่มี Link เข้ามาเชื่อมต่อกำหนดให้เป็นศูนย์

- สัดส่วนการเลี้ยวออกจาก Link เป็นค่าร้อยละของยวดยานที่เลี้ยวซ้าย ตรง และเลี้ยวขวาออกจาทางแยก โดยกำหนดเป็นเลขจำนวนเต็ม ผลรวมทั้ง 3 ทิศทางต้องมีค่าเท่ากับ 100 ในกรณีที่ไม่มี Link ที่มาเชื่อมต่อในทิศทางนั้น กำหนดให้มีค่าเท่ากับศูนย์

การจำลองทางแยก เป็นการจำลองระบบควบคุมสัญญาณไฟเป็นส่วนใหญ่ โดยทำหน้าที่เสมือนเป็นสวิทช์เชื่อมต่อ Link เข้าด้วยกันตามจังหวะเวลาที่กำหนด ข้อมูลที่ผู้ใช้ต้องกำหนดให้โปรแกรมคือ

- ระบบควบคุมของสัญญาณไฟที่ใช้ว่าเป็นแบบใด (Fixed Time หรือ Actuated)
- ช่วงเวลาของจังหวะสัญญาณไฟเขียว รอบเวลาสัญญาณไฟ
- จังหวะใดให้ยวดยานในทิศทางได้ผ่านทางแยกไปได้ และในกรณีที่กำหนดสัญญาณไฟควบคุมแบบ Actuated นั้นต้องกำหนดข้อมูลใหม่ของ Detector ด้วย เนื่องจากการทำงานของสัญญาณไฟประเภทนี้อาศัย Detector เป็นตัวตรวจวัดปริมาณจราจรเพื่อส่งผลการตรวจวัดมาให้ส่วนควบคุมสัญญาณไฟ

3.2.2 การจำลองยวดยานเข้าสู่โครงข่าย

การจำลองการเกิดยวดยานเข้าสู่ระบบโครงข่ายถนน ทำโดยการกำหนดค่า Time Headway ขึ้น โดยคำนวณจากข้อมูลปริมาณการจราจรที่ผู้ใช้กำหนดให้กับ Input Link ค่า Time Headway ที่โปรแกรมสร้างขึ้นมาไว้ มีรูปแบบการกระจาย 2 แบบคือ แบบสม่ำเสมอ (Uniform) และแบบสุ่ม (Random) ในกรณีแบบสม่ำเสมอนั้น ยวดยานที่เข้ามาจะมีค่า Time Headway เท่ากัน หรือเกิดยวดยานที่ระยะเวลาห่างเท่าๆ กัน แล้วโปรแกรมจะนำไปคำนวณเป็นเวลาที่มียวดยานเกิดขึ้นมา เมื่อโปรแกรมทำงานไปครบ 1 รอบ เวลาจะถูกเพิ่มขึ้น 1 วินาที โปรแกรมส่วนที่ทำหน้าที่จำลองการเกิดยวดยาน จะคอยตรวจสอบดูว่าเวลาของโปรแกรมในรอบปัจจุบันนั้นมากกว่าหรือเท่ากับ เวลาที่กำหนดให้มียวดยานเกิดขึ้นหรือไม่ ถ้าใช่ก็จะกำหนดให้ยวดยานนั้นเข้าสู่โครงข่ายใน block สุดท้ายของ Input Link

ส่วนกรณีของยวดยานที่เข้ามาแบบสุ่ม แตกต่างจากแบบแรกตรงที่ Time Headway จะมีค่าไม่เท่ากัน (หรือมียวดยานเกิดขึ้นในระยะเวลาห่างไม่เท่ากัน) โปรแกรมทำการสร้างลำดับค่า Time Heaway ขึ้นมาเป็นชุด เท่ากับยวดยานที่ผู้ใช้โปแกรมกำหนดให้เข้าสู่ Input Link แล้วจึงค่อยๆนำเอาค่า Time Headway ชุดนี้ไปใช้เป็นเช่นเดียวกับกรณีของแบบสม่ำเสมอคือนำไปคำนวณเวลาที่ยวดยานคันถัดไปจะเข้ามาสู่ Input Link

การจำลองการเกิดยวดยานโดยวิธีดังนี้ จะเกิดกรณีที่ยวดยานนั้นเข้ามาสู่ Link ไม่ลงตัวกับ Scanning Time Interval เช่น 10.68 วินาที เป็นต้น ในกรณีเช่นนี้ โปรแกรมจะจัดให้ยวดยานคันนั้นเข้าสู่ Link ในวินาทีถัดไปคือ วินาที่ 11 การที่ต้องสร้างค่า Time Headway ให้เป็นเป็นตัวเลขทศนิยม เนื่องจากต้องการควบคุมให้ยวดยานเข้ามาในช่วงเวลาที่กำหนดเท่ากับที่ผู้ใช้กำหนด

ปริมาณยวดยานที่เข้าสู่โครงข่ายถนน สามารถกำหนดให้เปลี่ยนแปลงตามช่วงเวลาได้ เช่น เปลี่ยนแปลงทุกๆ 15 นาที (900 วินาที) เป็นต้น หรืออาจจะกำหนดให้คงที่ตลอดช่วงเวลาที่ทำการทำงานของโปแกรม โดยกำหนดให้อยู่ในหน่วย คันต่อชั่วโมง

3.2.3 การเคลื่อนตัวของยวดยานภายใน Link

เมื่อมียวดยานเกิดขึ้นที่ block สุดท้ายของ Input Link แล้ว ยวดยานคันดังกล่าว หรือกลุ่มดังกล่าวจะถูกดำเนินการต่อไปในลักษณะของกลุ่มยวดยาน โดยกำหนดให้การเคลื่อนตัวของยวดยาน ขึ้นอยู่กับค่าดังต่อไปนี้

- ค่าอัตราการไหลอิ่มตัว (Saturation Flow Rate) เป็นค่าที่กำหนดเพดานสูงสุดที่ยวดยานสามารถเคลื่อนย้ายจาก block หนึ่งที่อยู่ด้าน Upstream ไปสู่อีก block หนึ่งที่อยู่ทางด้าน Downstream ได้ ค่านี้เป็นค่าที่คำนวณจากคุณสมบัติของ Link เมื่อนำมาใช้จะถูกเปลี่ยนให้อยู่ในหน่วย คัน/วินาที โดยคำนวณจากสูตร

อัตราการไหลอิ่มตัวที่ใช้ = อัตราการไหลอิ่มตัวต่อช่องทาง (หน่วย คัน/ช่องทาง/วินาที)
x จำนวนช่องทาง

- ค่าความจุของ Block ที่สามารถบรรจุยวดยานได้ ค่านี้เป็นค่าที่กำหนดเพดานของปริมาณยวดยานที่ Block หนึ่งๆ จะมีได้ในสภาพหยุดนิ่ง ค่านี้คำนวณจาก

สูตร

จำนวนยวดยานสูงสุดที่บรรจุได้ใน Block = ค่าความหนาแน่นสูงสุด (หน่วย คัน/เมตร/ช่องทาง) x จำนวนช่องทาง x ความยาวของ block (หน่วย เมตร)

ค่าของความหนาแน่นสูงสุด (Jam Density) เป็นค่าคงที่ในโปรแกรม ซึ่งมีผลกับทุก Link กำหนดไว้เท่ากับ 0.143 คัน/เมตร/ช่องทาง หรือประมาณ 7 เมตร/คัน จากค่าพารามิเตอร์ที่กล่าวมาจะเห็นว่า ความจุของ Link ขึ้นอยู่กับจำนวนช่องทางโดยตรง ถ้าจำนวนช่องทางมาก ความจุของ Link ก็มากตามไปด้วย ถ้าจำนวนช่องทางน้อย ความจุของ Link ก็น้อยด้วย

การเคลื่อนที่ของยวดยานระหว่าง block นั้น โปรแกรมจะพิจารณาการเคลื่อนที่ของยวดยานในแต่ละ block เป็นคู่ๆ ไป โดยเริ่มจาก block สุดท้ายของ Link กับ block ที่อยู่ถัดมาตามทิศทางกระแสการจราจร (Downstream) 1 block จนถึง block แรกกับ block ที่สอง การพิจารณาดูจากปริมาณยวดยานที่มีอยู่ใน block ร่วมกับสถานะของ block โดยกำหนดสภาถานะของ block ไว้ 3 สถานะคือ

สถานะ 0 คือ สถานะที่ block มีปริมาณยวดยานตั้งแต่ศูนย์ขึ้น แต่น้อยกว่า Jam Density

สถานะ 1 คือ สถานะที่ block มีปริมาณยวดยานเท่ากับค่า Jam Density ซึ่งหมายความว่ายวดยานทุกคันใน block นั้นหยุดหมด

สถานะ 2 คือ สถานะที่มียวดยานเคลื่อนที่ออกจาก block ในวินาทีที่แล้ว โดยสถานะในวินาทีที่แล้วนั้นเป็น 1 หรือ 2 หมายความว่าในขณะนั้น block นั้นมีสภาพเป็นคลื่นออกตัวจากหยุดนิ่ง (Starting wave)

โปรแกรมจะทำการคำนวณค่าปริมาณยวดยานที่สามารถเคลื่อนที่ไปอยู่ใน block ต่อไปได้ โดยตรวจสอบเนื้อที่ว่างใน block ด้าน Downstream แล้วเปรียบเทียบกับปริมาณยวดยานที่มีอยู่ใน block ทางด้านย้อนกระแสการจราจร (Upstream) ตามสมการดังนี้

$AF_{K,t}$ = ค่าน้อยที่สุดของ

SFRT - NLANE หรือ

$EV_{K+1,t-1}$ หรือ

$$JDS \cdot UL \cdot NLANE - EV_{K,t-1}$$

โดยที่ $AF_{K,t}$ = ปริมาณยวดยานที่เคลื่อนที่เข้ามาใน block ที่ K (block ที่กำลังพิจารณาอยู่) ในเวลาปัจจุบันคือ วินาทีที่ t หน่วยเป็นคัน

$EV_{K+1,t-1}$ = ปริมาณยวดยานที่อยู่ใน block ที่ K+1 หรือ block ที่อยู่ทางด้าน Upstream ในวินาทีที่แล้ว หน่วยเป็นคัน

$EV_{K,t-1}$ = ปริมาณยวดยานที่อยู่ใน block ที่ K ในวินาทีที่แล้ว หน่วยเป็นคัน

SFRT = อัตราการไหลอิ่มตัว หน่วย คัน/ช่องทาง/วินาที

NLANE = จำนวนช่องทางของ Link

JDS = ค่า Jam Density หน่วย คัน/เมตร/ช่องทาง

UL = ความยาวของ block หน่วยเป็นเมตร

ค่า $AF_{K,t}$ ที่คำนวณได้จะถูกนำไปใช้คำนวณปริมาณยวดยานที่บรรจุอยู่ใน block แต่ละ block แล้ว นำเอาค่าปริมาณยวดยานที่ได้ไปคำนวณสถานะของ block ตามเงื่อนไขดังแสดงในรูปที่ 3.8

3.2.4 การเคลื่อนตัวของยวดยานระหว่าง Link

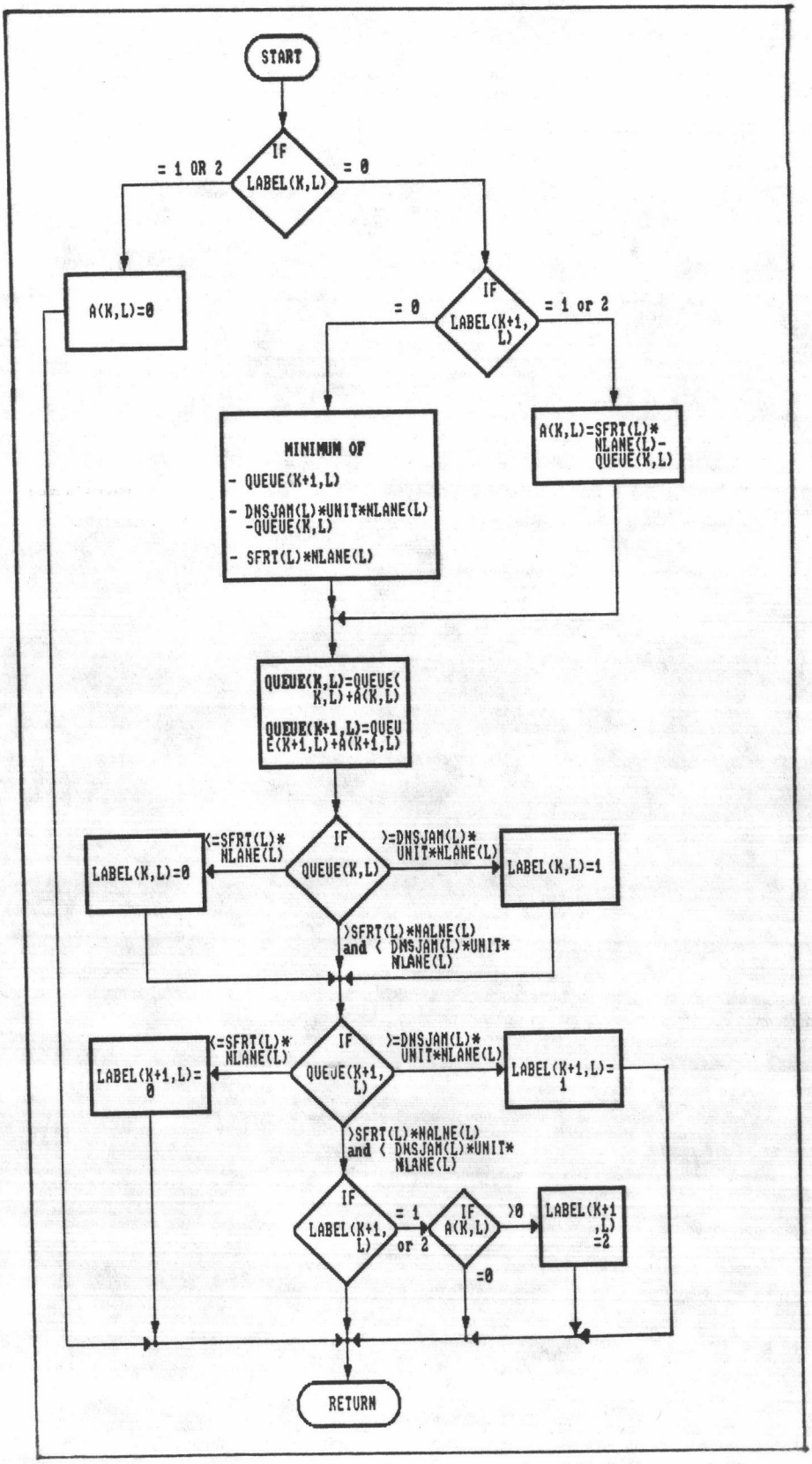
โปรแกรมย่อยที่ดำเนินการคือ OTFLOW โดยพิจารณาว่า Link นั้นเป็น Link ประเภทใด ถ้าเป็น Link ออกจากโครงข่ายก็จะทำการลบปริมาณยวดยานที่บรรจุอยู่ใน block แรกออก แต่ถ้าเป็น Link ที่เข้าสู่ทางแยก จะดำเนินการตรวจสอบจังหวะสัญญาณไฟในขณะนั้นว่า ให้ยวดยานเลี้ยวไปในทิศทางใดได้บ้าง โดยเริ่มจากยวดยานที่เลี้ยวซ้าย ทิศทางตรง และเลี้ยวขวา กรณีของยวดยานที่เลี้ยวขวานั้น แยกเป็นกรณีที่เลี้ยวขวาขณะมียวดยานสวนทางมาหรือไม่ ถ้าเลี้ยวขวาโดยไม่มียวดยานสวนทางมาก็ให้เลี้ยวขวาตามปกติ แต่ถ้ามีก็ให้เลี้ยวขวาโดยพิจารณาหาช่องว่างระหว่างยวดยานที่สวนทางมา

ปริมาณยวดยานที่เคลื่อนที่ไปสู่ Link ต่อไปนั้น พิจารณาจากดังนี้

ปริมาณยวดยานที่เคลื่อนตัวออกจาก Link = ค่าต่ำสุดของ

อัตราการไหลอิ่มตัวของ Link x จำนวนช่องทางใน Link

หรือเท่ากับ ความจุสูงสุดของ Block สุดท้ายใน Link ที่จะเคลื่อนไป - ปริมาณยวดยาน



รูปที่ 3.8 ผังการทำงานในส่วนพิจารณาการเคลื่อนตัวของยวดยานภายใน Link

ที่บรรจุอยู่ใน Block สุดท้ายของ Link ที่จะเคลื่อนไป
หรือเท่ากับ ปริมาณยวดยานที่อยู่ใน Block แรกของ Link

การเคลื่อนตัวของยวดยานที่เลี้ยวขวาออกจากทางแยกเป็นแบบ Microscopic คือ ออกจากทางแยกเป็นคันๆ โดยแยกพิจารณาเป็นกรณีๆ ดังนี้

- ในกรณีที่ไม่มีการกำหนดช่องทางสำหรับเลี้ยวขวาไว้โดยเฉพาะ โปรแกรมจะรวบรวมยวดยานที่ออกจาก Link ถ้ามีค่ามากกว่าหรือเท่ากับ 1 คัน ให้ยวดยานออกไปได้เท่ากับเลขหลักจำนวนเต็ม แต่ถ้ามีปริมาณยวดยานน้อยกว่า 1 คัน ให้นำไปเปรียบเทียบกับเลขสุ่ม ถ้าเลขสุ่มมีค่าน้อยกว่าให้ยวดยานออกไปได้ 1 คัน ถ้ามากกว่าให้เก็บค่าปริมาณยวดยานไว้รวมกับยวดยานที่เลี้ยวขวาที่จะเข้ามาในวินาทีถัดไป

- กรณีที่มีการจัดช่องทางไว้สำหรับยวดยานที่เลี้ยวขวาโดยเฉพาะ ยวดยานที่ต้องการออกจากทางแยกจะถูกรวบรวมจนกระทั่งมีค่าตั้งแต่ 1 คันขึ้นไปจึงจะถูกปล่อยออกไปวินาทีละ 1 คัน แต่ถ้ายวดยานที่ต้องการเลี้ยวขวามีค่าน้อยกว่า 1 คัน จะไม่มีการปล่อยยวดยานออกจากทางแยกโดยการนำปริมาณยวดยานไปเปรียบเทียบกับกรณีที่มีไม่การจัดช่องเลี้ยวขวาไว้โดยเฉพาะ

จากรูปที่ 3.8 ความหมายของตัวแปรแต่ละตัวมีดังนี้

LABEL(K,L) = สถานะของ block ที่ k Link ที่ L
A(K,L) = ปริมาณยวดยานที่เคลื่อนที่เข้ามาใน block ที่ K Link ที่ L หน่วยเป็นคัน
QUEUE(K,L) = ปริมาณยวดยานที่บรรจุอยู่ใน block ที่ K Link ที่ L หน่วยเป็นคัน
SFRT(L) = อัตราการไหลอิมตัวของ Link ที่ L เป็นหน่วยคันต่อวินาที
DNSJAM(L) = ความหนาแน่นสูงสุดที่ Block หนึ่งๆใน Link ที่ L สามารถบรรจุอยู่ได้ หน่วยเป็นคันต่อเมตรต่อ 1 ช่องทาง
NLANE(L) = จำนวนช่องทางของ Link L

จากสมการของ block ที่กำหนดขึ้น แสดงว่าความเร็วของยวดยานมี 2 ค่าเท่านั้น คือ ความเร็วอิสระ (Free Flow Speed) และความเร็วเป็นศูนย์

3.2.5 การเกิดคิวของยวดยาน

แบบจำลองประเภท Macroscopic Model บางโปรแกรมนั้นจำลองการเกิดคิวของยวดยานเป็นแบบที่เรียกว่า " Piling Up Queue " หรือ " Vertical Queue " คือ ยวดยานจะจอดรอสัญญาณไฟที่บริเวณเส้นหยุดทุกคัน ทำให้พฤติกรรมการเกิดคิวแตกต่างจากพฤติกรรมที่เกิดขึ้นจริงมาก ความยาวของคิวคำนวณจากจำนวนยวดยานคูณกับความยาวเฉลี่ยของยวดยานแต่ละคัน อีกทั้งไม่ได้คิดผลกระทบ เนื่องจากความยาวคิวต่อระยะทางของ Link ที่ยวดยานด้าน Upstream สามารถเคลื่อนตัวเข้ามาจอดท้ายคิว ยวดยานที่เข้ามาต่อคิวจึงสามารถเคลื่อนตัวเข้ามาจอดท้ายคิวที่เส้นหยุดได้เลย การจำลองแบบนี้ไม่สามารถนำไปใช้ในการจำลองการติดตั้ง detector ได้ เนื่องจากการตรวจวัดสภาพการจราจรจาก detector จะไม่สะท้อนผลกระทบจากความยาวคิวที่ยาวมาทับตำแหน่งที่มีการติดตั้ง detector

ใน ซี ยู ทราฟฟิค ซิมูเลชั่น โปรแกรม นี้ได้ทำการจำลองโครงข่ายถนนแบ่งเป็นส่วนๆ ซึ่งเรียกว่า " block " ในแต่ละ block นั้นมีการตรวจสอบสถานะเสมอว่ามียวดยานเคลื่อนตัว หรือหยุดอยู่ จึงสามารถประมาณความยาวคิวที่เกิดขึ้นได้ แต่ความละเอียดนั้นขึ้นอยู่กับความยาวของ block ที่กำหนดขึ้น

ความยาวคิวของแต่ละ Link วัดจากเส้นหยุดจนถึงยวดยานคันสุดท้ายที่มาต่อท้ายคิว โดยเริ่มตรวจสอบจาก block ท้ายสุดทางด้าน Upstream มาทางด้าน Downstream จนพบ block ที่มีสถานะเป็น 1 แล้วตรวจสอบ block ถัดไปทางด้าน Upstream อีก 1 block ว่ามียวดยานอยู่ใน block เท่าใด แล้วคำนวณความยาวคิวของยวดยานจนถึง block นั้น โดยคิดความยาวคิว ดังนี้

ความยาวคิว (เมตร) = (ตำแหน่งของ block ท้ายสุดที่มีสถานะเป็น 1 x ความยาวของ block) + (จำนวนยวดยานใน block ถัดไปทางด้าน Upstream 1 block x ความยาวของ block / จำนวนยวดยานสูงสุดที่สามารถบรรจุใน block ได้)

จากวิธีการนี้ทำให้การจำลองการจราจรในขณะที่เกิดคิว และการเคลื่อนตัวจากสภาพหยุดที่ก่อตัวเป็นคิว ใกล้เคียงกับสภาพจริงมากกว่าการจำลองการเกิดคิวแบบ Piling Up ที่เส้นหยุด

3.2.6 การตรวจวัดสภาพการจราจรโดย detector

เนื่องจากอุปกรณ์ที่ใช้ในการตรวจวัดสภาพการจราจรจริงๆนั้น จะทำงานเมื่อมียวดยานผ่านมาเป็นคันๆ ไป แต่เนื่องจากแบบจำลองการจราจรเป็นแบบ Macroscopic ซึ่งจำลองกลุ่มการไหลของยวดยาน จึงไม่สามารถระบุการเคลื่อนที่ของยวดยานเป็นคันๆ ได้ ดังนั้นจึงต้องมีกระบวนการที่สามารถระบุตำแหน่งของยวดยานเป็นคันๆจากกลุ่มการไหลการจราจรที่จำลองขึ้น ดังขั้นตอนดังต่อไปนี้

- ทุกๆ Scanninng time Interval (ทุกๆ 1 วินาที) คำนวณค่าสะสมของปริมาณยวดยานใน block ที่มีการติดตั้ง detector

- ตรวจสอบค่าสะสมดังกล่าวทุกๆ Scanning Time Interval ถ้ามีค่ามากกว่าหรือเท่ากับให้ผลการตรวจวัดปริมาณการจราจรได้ 1 คัน แล้วลบค่าสะสมออกด้วย 1 คัน

- ในกรณีที่ค่าสะสมของยวดยานนั้นมีค่าอยู่ระหว่าง 0 ถึง 1 ให้นำค่านี้ไปเปรียบเทียบกับเลขสุ่ม (ซึ่งมีค่าอยู่ตั้งแต่ 0 ถึง 1) ถ้าค่าสะสมของยวดยานมากกว่า ก็ให้ผลตรวจวัดปริมาณการจราจรได้ 1 คัน แล้วทำการลบค่าสะสมของยวดยานออกด้วย 1 คัน

- ในกรณีที่ค่าสะสมของยวดยานเท่ากับหรือน้อยกว่าศูนย์ ให้ผลการตรวจวัดปริมาณการจราจรเท่ากับศูนย์

ใน ซี ยู ทราฟฟิค ซิมูเลชั่น โปรแกรม มีโปรแกรมย่อย DETECT เป็นโปรแกรมย่อยที่ทำหน้าที่จำลองการตรวจวัดสภาพการจราจรด้วย detector

สิ่งที่จะได้จากการตรวจวัดสภาพการจราจรโดย Detector อีกอย่างหนึ่งคือ เวลาที่ตรวจวัดยวดยาน แต่จากขั้นตอนที่ผ่านมานั้นเวลาที่ตรวจวัดได้มีความละเอียด (accuracy) เท่ากับ Scanning Time Interval เท่านั้น การที่จะทำให้ความละเอียดสูงขึ้นต้องอาศัยตัวเลขสุ่มมาช่วยคำนวณดังนี้

3.2.7 การคำนวณเวลาที่ยวดยานผ่าน detector

ประเภทของ detector ที่ใช้มีหลายประเภท แต่มี detector บางประเภทที่สามารถให้ผลการตรวจสภาพการจราจรเป็นตัวแปรอีกแบบหนึ่ง นอกจากปริมาณการจราจรที่ตรวจวัดได้เพียงอย่างเดียว ตัวแปรนั้นคือ ค่า Time Occupancy ของยวดยาน detector ที่สามารถให้ค่านี้ได้เป็น detector ประเภท Loop detector หรือแบบ Ultrasonic detector

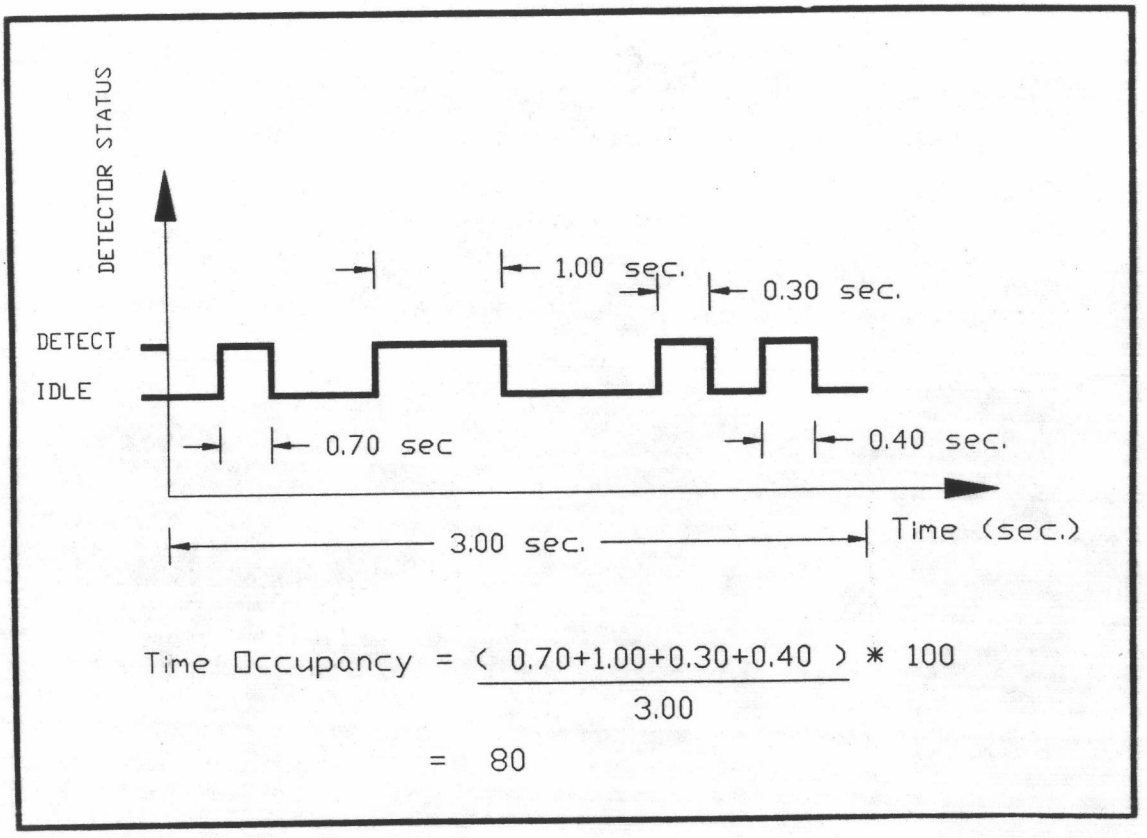
ค่า Time Occupancy คือ อัตราส่วนระหว่างเวลาที่ยวดยานวิ่งผ่าน detector ต่อช่วงเวลาที่ทำการตรวจวัด ซึ่งนิยมวัดออกมาเป็นเปอร์เซนต์ ค่า Time Occupancy นี้ถือเป็นตัวแปรตัวหนึ่งที่สามารถสะท้อนสภาพการจราจรได้เช่นเดียวกับ ปริมาณการจราจร ความหนาแน่น และความเร็ว

การวัดค่า Time Occupancy ของ detector นั้นเริ่มจาก detector จะส่งสัญญาณออกไปเป็นช่วงสั้นๆ (Pulse) ช่วงละเท่าๆกัน โดยเช่น ประมาณ 0.1 วินาที เป็นต้น ในขณะที่ไม่มียวดยานผ่านมา สัญญาณที่ได้รับกลับมาให้มีสถานะเป็นศูนย์หรือ Off-Pulse ในกรณีที่มียวดยานผ่านมาสถานะเป็น On-Pulse เมื่อรวมระยะเวลา On-Pulse ที่เกิดขึ้น แล้วหารด้วยช่วงเวลาที่ผ่านมาทั้งหมด จะได้เป็นค่า Time Occupancy การคำนวณแสดงในรูปที่ 3.9

การทำงานของโปรแกรมย่อย detect นี้จะไปเรียกใช้โปรแกรมย่อย PULS1 ซึ่งทำหน้าที่จำลองการคำนวณค่า Pulse Length ได้จากการตรวจวัดโดย detector โดยคำนวณจากสูตร

$$\text{Pulse Length} = \frac{(\text{ความยาวของยวดยาน} + \text{ความยาวของ detector})}{\text{ความเร็วที่ยวดยานวิ่งผ่าน}}$$

ความยาวของยวดยาน เป็นความยาวประสิทธิผล โดยวัดจาก ล้อหน้าถึงล้อหลัง ในกรณีที่ใช้ detector แบบขดลวดเหนี่ยวนำ (Loop Detector) ถ้าเป็น detector แบบ Ultrasonic วัดจากกันชนหน้าถึงกันชนหลัง ความยาวของพื้นที่ตรวจวัดประมาณ 3 เมตร จากสูตรดังกล่าวจะเห็นว่า Pulse Length ขึ้นอยู่กับ ความเร็วที่ยวดยานวิ่งผ่านเท่านั้นถ้าให้ตัวแปรอื่นๆ คงที่ ถ้าความเร็วของยวดยานต่ำ Pulse Length ที่คำนวณได้จะมีค่ามาก แต่ถ้าความเร็วสูง Pulse Length ที่คำนวณได้จะมีค่าน้อย โดยที่กำหนดให้ความยาวของยวดยาน



$$\text{Time Occupancy} = \frac{(\ 0.70+1.00+0.30+0.40\ )}{3.00} * 100$$

$$= 80$$

รูปที่ 3.9 รูปแสดงการคำนวณค่าเฉลี่ยของ Time Occupancy

เท่ากัน

การคำนวณ Pulse Length ในแบบจำลอง ซี ยู ทราฟฟิค ซิมูเลชั่น โปรแกรมไม่สามารถ คำนวณจากสูตรดังกล่าวได้โดยตรง เนื่องจากแบบจำลองไม่ได้มีการจำลองค่าความเร็วของยวดยานเป็นคันๆไป ดังนั้นจึงต้องคำนวณโดยประมาณจากตัวแปรทางด้านการจราจรที่มีอยู่ในแบบจำลอง ดังสูตรต่อไปนี้

$$\text{Pulse Length} = \frac{\text{APL} + (\text{USI}-\text{APL}) \times (EF_t - AF_t)}{\text{JAM FLOW}}$$

โดยที่

APL = ค่าเฉลี่ยของ Pulse Lenth (Average Pulse Length) มีหน่วยเป็น วินาที

USI = Unit Scanning Time Interval หรือช่วงเวลาที่เปลี่ยนไปเมื่อโปรแกรมทำงานงาน ครบหนึ่งรอบ มีค่าเท่ากับ 1 วินาที

JAM FLOW = จำนวนยวดยานที่ Block สามารถบรรจุได้สูงสุดใน 1 ช่องทาง ในที่นี้ใช้เท่ากับ 2 คัน

EF_t = จำนวนยวดยานที่มีอยู่ใน Block ที่มีการติดตั้ง detector เมื่อวินาทีที่ t

AF_t = จำนวนยวดยานที่เคลื่อนที่เข้ามาสู่ Block ที่มีการติดตั้ง detector เมื่อวินาทีที่ t

ค่า Average Pulse Length เป็นค่าที่เปลี่ยนไปทุกๆ 1 รอบเวลา โดยทำการสุ่มขึ้นมา มาลักษณะการกระจายเป็นแบบปกติ (Normal Distribution) โดยใช้ค่าเบี่ยงเบนมาตรฐานเท่ากับ 0.1 วินาที ส่วนค่าเฉลี่ยเลขคณิตจะใช้ค่าเริ่มต้น (Initial) เท่ากับ 0.6 วินาที แต่จะมีการคำนวณใหม่ทุกๆรอบเวลาสัญญาณไฟ โดยคำนวณจากค่า Pulse Length ทุกค่าที่ในรอบสัญญาณไฟรอบที่ผ่านมา