ຄຳນຳ

ປັນຍາປະດິດ ເປັນວິຊາໜຶ່ງທີ່ສຳຄັນ ໃນລາຍວິຊາຮຽນຂອງສາຂາວິທະຍາສາດຄອມພິວເຕີ ທີ່ສຶກສາກ່ຽວກັບ ຄວາມຮູ້ພື້ນຖານ, ປະຫວັດຄວາມເປັນມາ, ການປະຍຸກໃຊ້ປັນຍາປະດິດ, ການແກ້ປັນຫາດ້ວຍເຕັກນິກການຄົ້ນຫາ (Problem Solving using Search), ວິທີການສະແດງຄວາມຮູ້ (Knowladge Represetation), ການຫາຂໍ້ສະຫຼຸບພາຍໃຕ້ຄວາມບໍ່ແນ່ນອນ (Inference under Uncertainty), ການຮຽນຮູ້ຂອງເຄື່ອງ (Machine Learning), ເຄືອຂ່າຍເສັ້ນປະສາດທຽມ (Artificial Neuron Network) ແລະ ລະບົບຜູ້ຊ່ຽວຊານ (Expert System). ເພາະວິຊານີ້ເປັນພື້ນຖານຂອງການຮຽນສາຂາວິທະຍາສາດຄອມພິວເຕີ.

ດັ່ງນັ້ນ, ທາງຄູອາຈານພາຍໃນພາກວິຊາວິທະຍາສາດຄອມພິວເຕີ ຈຶ່ງໄດ້ຮຽບຮຽງເອກະສານເຫຼັ້ມນີ້ຂຶ້ນເພື່ອໃຫ້ນັກສຶກສາ ແລະ ຄູອາຈານ ໄດ້ມີປຶ້ມຕຳລາ ປະກອບເຂົ້າໃນການຮຽນ-ການສອນ ວິຊາປັນຍາປະດິດ ແລະ ເພື່ອເປັນເອກະສານຄົ້ນຄວ້າສຳລັບຜູ້ສົນໃຈໃນວິຊານີ້. ໂຄງສ້າງເນື້ອໃນຂອງປຶ້ມຫົວນີ້ ແມ່ນໄດ້ຮຽບຮຽງຕາມເນື້ອໃນຫຼັກສູດ ວິທະຍາສາດຄອມພິວເຕີ, ຄະນະວິທະຍາສາດທຳມະຊາດ, ມະຫາວິທະຍາໄລແຫ່ງຊາດ. ເຖິງວ່າເນື້ອໃນຂອງປຶ້ມຫົວນີ້ໄດ້ຜ່ານການກວດແກ້ແລ້ວກໍ່ຕາມ, ແຕ່ກໍຍັງປາສະຈາກໄດ້ຂໍ້ຂາດຕົກບົກພ່ອງ. ດັ່ງນັ້ນ, ພວກຂ້າພະເຈົ້າຈະຖືເປັນກຽດຢ່າງສູງ, ຖ້າທ່ານຜູ້ອ່ານໃຫ້ຄວາມກະລຸນາສົ່ງຂ່າວມາຫາພວກຂ້າພະເຈົ້າ, ເພື່ອເປັນການປັບປຸງປຶ້ມຫົວນີ້ໃຫ້ມີຄວາມຄົບຖ້ວນສົມບູນ ແລະ ດີຂຶ້ນເລື້ອຍໆ.

ພ້ອມນີ້ພວກຂ້າພະເຈົ້າຂໍສະແດງຄວາມຂອບໃຈ ແລະ ຮູ້ບຸນຄຸນຢ່າງສູງມາຍັງຄະນະນຳ ຂອງ ມະຫາວິທະຍາໄລແຫ່ງຊາດ, ຄະນະກຳມະການວິຊາການ, ຄະນະບໍດີຄະນະວິທະຍາສາດທຳມະຊາດ, ຄະນະພາກວິຊາວິທະຍາສາດຄອມພິວເຕີ ແລະ ທຸກໆທ່ານທີ່ສະໜັບສະໜູນການພິມປຶ້ມເຫຼັ້ມນີ້.

ນະຄອນຫຼວງວຽງຈັນ, ວັນທີ 16 ສິງຫາ 2018

ຄະນະຮຽບຮຽງ ແລະ ກວດແກ້

,

ສາລະບານ

ບົດທີ 1 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບປັນຍາປະດິດ (Introduction to Artificial Intelligence)

1.1 ປັນຍາປະດິດແມ່ນຫຍັງ?

ມີຄຳນິຍາມຂອງປັນຍາປະດິດ(Artificial Intelligence :AI) ຫຼວງຫຼາຍ ເຊິ່ງສາມາດຈັດແບ່ງອອກເປັນ 4 ປະເພດໂດຍເບິ່ງໃນ 2 ມິຕິ ໄດ້ແກ່: ລະຫວ່າງ ນິຍາມທີ່ເນັ້ນລະບົບທີ່ຮຽນແບບມະນຸດ ກັບ ນິຍາມທີ່ເນັ້ນລະບົບທີເປັນລະບົບທີມີເຫດຜົນ (ແຕ່ບໍ່ຈຳເປັນຕ້ອງຄືມະນຸດ), ລະຫວ່າງ ນິຍາມທີ່ເນັ້ນຄວາມຄິດເປັນຫຼັກ ກັບ ນິຍາມທີ່ເນັ້ນການເຮັດວຽກເປັນຫຼັກ

ປັດຈຸບັນງານວິໄຈຫຼັກໆ ຂອງ AI ຈະມີແນວຄິດໃນຮູບທີ່ເນັ້ນເຫດຜົນເປັນຫຼັກ ເນື່ອງຈາກການນຳ AI ໄປປະຍຸກໃຊ້ແກ້ບັນຫາ ບໍ່ຈຳເປັນຕ້ອງອາໄສອາລົມ ຫຼື ຄວາມຮູ້ສຶກຂອງມະນຸດ ຢ່າງໃດກໍຕາມນິຍາມທັງ 4 ບໍ່ໄດ້ຕ່າງກັນໂດຍສົມບູນ ນິຍາມທັງ 4 ຕ່າງກໍມີສ່ວນຮ່ວມທີ່ກ່ຽວກັນຢູ່

1. ລະບົບທີ່ຄິດຄືມະນຸດ (Systems that think like humans)

- [AI ແມ່ນ] ຄວາມພະຍາຍາມໃໝ່ອັນໜ້າຕື່ນເຕັ້ນທີ່ຈະເຮັດໃຫ້ຄອມພິວເຕີຄິດໄດ້ … ເຄື່ອງຈັກທີມີສະຕິປັນຍາຢ່າງຄົບຖ້ວນແລະແທ້ຈິງ (“The exciting new effort to make computers think … machines with minds, in the full and literal sense.” [Haugeland, 1985])
- [AI ແມ່ນ ກິນໄກຂອງ] ກິດຈະກຳທີ່ກ່ຽວຂ້ອງກັບຄວາມຄິດມະນຸດ, ເຊັ່ນ: ການຕັດສິນໃຈ, ການແກ້ບັນຫາ, ການຮຽນຮູ້ (“[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning.”[Bellman, 1978])

2. ລະບົບທີເຮັດວຽກຄືມະນຸດ (Systems that act like humans)

- [AI ແມ່ນ] ວິຊາຂອງການສ້າງເຄື່ອງຈັກລະທີ່ເຮັດວຽກໃນສິ່ງເຊິ່ງອາໄສປັນຍາເມື່ອເຮັດວຽກໂດຍມະນຸດ (*“The art of creating machines that perform functions that requires intelligence when performed by people.”*[Kurzweil, 1990])
- [AI ແມ່ນ] ການສຶກສາວິທີເຮັດໃຫ້ຄອມພິວເຕີເຮັດວຽກໃນສິ່ງທີ່ມະນຸດເຮັດໄດ້ດີກວ່າໃນຂະນະນັ້ນ (*“The study of how to make computers do things at which, at the moment, people are better.”*[Rich and Knight, 1991])

*ໝາຍເຫດ ການເຮັດວຽກຄືມະນຸດ, ເຊັ່ນ:

- ສື່ສານໄດ້ດ້ວຍພາສາທີ່ມະນຸດໃຊ້, ເຊັ່ນ: ພາສາລາວ, ພາສາອັງກິດ. ຕົວຢ່າງ: ການແປງຂໍ້ຄວາມເປັນຄຳເວົ້າ ແລະ ການແປງຄຳເວົ້າເປັນຂໍ້ຄວາມ.
- ມີປະສາດຮັບສຳພັດຄ້າຍຄືມະນຸດ, ເຊັ່ນ: ຄອມພິວເຕີຮັບພາບໄດ້ໂດຍອຸປະກອນຮັບສຳພັດ ແລ້ວເອົາພາບໄປປະມວນຜົນ.
- ເຄື່ອນໄຫວໄດ້ຄ້າຍຄືມະນຸດ, ເຊັ່ນ: ຫຸ່ນຍົນຊ່ວຍວຽກຕ່າງໆ. ຕົວຢ່າງ: ການດູດຝຸ່ນ, ເຄື່ອນຍ້າຍສິ່ງຂອງ.

- ຮຽນຮູ້ໄດ້ໂດຍສາມາດກວດຈັບຮູບແບບການເກີດຂອງເຫດການໃດໆ, ແລ້ວປັບຕົວສູ່ສິ່ງແວດລ້ອມທີ່ປ່ຽນໄປໄດ້.

3. ລະບົບທີ່ຄິດຢ່າງມີເຫດຜົນ (Systems that think rationally)

- [AI ແມ່ນ] ການສຶກສາຄວາມສາມາດໃນດ້ານສະຕິປັນຍາໂດຍການໃຊ້ແບບຈຳລອງການຄິດໄລ່ ("The study of mental faculties through the use of computational model." [Charniak and McDermott, 1985])
- [AI ແມ່ນ] ການສຶກສາວິທີການຄິດໄລ່ທີ່ສາມາດຮັບຮູ້, ໃຊ້ເຫດຜົນ ແລະ ເຮັດວຽກ ("The study of the computations that make it possible to perceive, reason, and act" [Winston, 1992])

  *ໝາຍເຫດ ຄິດຢ່າງມີເຫດຜົນ ຫຼື ຄິດຖືກຕ້ອງ, ເຊັ່ນ: ໃຊ້ຫຼັກການຕັກສາດໃນການຄິດຫາຄຳຕອບຢ່າງມີເຫດຜົນ, ເຊັ່ນ: ລະບົບຜູ້ຊ່ຽວຊານ.

4. ລະບົບທີ່ເຮັດວຽກຢ່າງມີເຫດຜົນ (Systems that act rationally)

- ປັນຍາປະດິດແມ່ນການສຶກສາເພື່ອອອກແບບຕົວເຮັດວຽກທີ່ມີປັນຍາ ("Computational Intelligence is the study of the design of intelligent agents" [Poole et al., 1998])
- AI ກ່ຽວຂ້ອງກັບພຶດຕິກຳທີ່ສະແດງປັນຍາໃນສິ່ງທີ່ມະນຸດສ້າງຂຶ້ນ ("AI … is concerned with intelligent behavior in artifacts" [Nilsson, 1998])

  * ໝາຍເຫດ ເຮັດວຽກຢ່າງມີເຫດຜົນ ເຊັ່ນ Agent (ໂປຣແກຣມທີ່ມີຄວາມສາມາດໃນການເຮັດວຽກ ຫຼື ເປັນຕົວແທນໃນລະບົບອັດໂນມັດຕ່າງໆ) ສາມາດເຮັດວຽກຢ່າງມີເຫດຜົນເພື່ອບັນລຸເປົ້າໝາຍທີ່ໄດ້ຕັ້ງໄວ້, ເຊັ່ນ: Agent ໃນລະບົບຂັບລົດອັດໂນມັດ ທີ່ມີເປົ້າໝາຍວ່າຕ້ອງໄປເຖິງເປົ້າໝາຍໃນໄລຍະທາງທີ່ສັ້ນທີ່ສຸດ, ຕ້ອງເລືອກເສັ້ນທາງທີ່ໄປຫາເປົ້າໝາຍທີ່ສັ້ນທີ່ສຸດທີ່ເປັນໄປໄດ້, ຈຶ່ງຈະເອີ້ນໄດ້ວ່າ Agent ເຮັດວຽກຢ່າງມີເຫດຜົນ, ຕົວຢ່າງເຊັ່ນ: Agent ໃນເກມໝາກລຸກ ທີ່ມີເປົ້າໝາຍວ່າຕ້ອງເອົາຊະນະຄູ່ຕໍ່ສູ້ ກໍຕ້ອງເລືອກຍ່າງໝາກທີ່ຈະເຮັດໃຫ້ຄູ່ຕໍ່ສູ້ພ່າຍແພ້ໃຫ້ໄດ້ ເປັນຕົ້ນ.

❖ **Turing Test**

ຍ້ອນກັບຄືນໄປໃນປີ 1950 ຫຼັງຈາກຄອມພິວເຕີໄດ້ຖືກຄົ້ນຄິດຂຶ້ນ Alan Turing ໄດ້ສະເໜີການທົດສອບປັນຍາຂອງເຄື່ອງຄອມພິວເຕີ ແລະ ໄດ້ກາຍເປັນທີ່ຮູ້ຈັກໃນນາມ ການທົດສອບທົວຣິງ (Turing Test), ໃນການທົດສອບນີ້ ຄອມພິວເຕີ ແລະ ມະນຸດໄດ້ ສົນທະນາກັບຜູ້ທົດສອບເປັນເວລາ 5 ນາທີ ແລ້ວຈາກນັ້ນຜູ້ທົດສອບໄດ້ຕັດສິນໃຈວ່າອັນໃດແມ່ນຄົນ ອັນໃດແມ່ນຄອມ. ມີການແຂ່ງຂັນປະຈຳປີເຊິ່ງເອີ້ນວ່າ the Loebner Prize, ເປັນການແຂ່ງໂປຣແກຣມທີ່ຄ້າຍຄືມະນຸດ, ສຳລັບໂປຣແກຣມທີ່ເຂົ້າແຂ່ງຂັນ ບໍ່ມີໂປຣແກຣມໃດທີ່ເຮັດໄດ້ເທົ່າກັບ Turing Predicted. ແຕ່ພວກເຂົາກໍ່ເຮັດໄດ້ດີຂຶ້ນທຸກຄັ້ງ ເຊັ່ນດຽວກັນກັບ ໂປຣແກຣມໝາກລຸກທີ່ສາມາດຊະນະ grandmasters, ຄອມພິວເຕີຈະເວົ້າໄດ້ຢ່າງຄ່ອງແຄ້ວຄືກັບມະນຸດ. ເມື່ອເຮັດແນວນັ້ນຈະເຫັນໄດ້ວ່າທັກສະການສົນທະນາບໍ່ແມ່ນການພິສູດຄວາມສະຫຼາດ. ນັ້ນແມ່ນປັນຫາເພາະເຮົາບໍ່ສາມາດຮູ້ໄດ້ວ່າມີໃຜຄິດແນວໃດແດ່, ການສື່ສານເປັນສິ່ງທີ່ເຮົາມີ.

1.2 ປະຫວັດຄວາມເປັນມາຂອງປັນຍາປະດິດ (History of AI)

<u>ກ່ອນກຳເນີດ AI</u> (ຄ. ສ. 1943-1955)

-ໂປຣແກຣມປັນຍາປະດິດເປັນໂປຣແກຣມທຳອິດໃນການຈຳລອງໜ່ວຍປະສາດດ່ຽວ (neuron) ສ້າງ ໂດຍ Warren McCulloch ແລະ Walter Pits ໂດຍໃຊ້ຄວາມຮູ້ເລື່ອງໜ້າທີ່ຂອງສະໝອງໃນດ້ານການ ພາບ, ຕັກສາດ. ທິດສະດີການຄິດໄລ່ ແລະ ພາຍຫຼັງ Donald Hebb ໄດ້ສະເໜີກົດການຮຽນຮູ້ເພື່ອ ອະທິບາຍການຮຽນຮູ້ຂອງເຄືອຂ່າຍເສັ້ນປະສາດທຽມ

- ມາລ໌ວິນ ມິນສກີ (Marvin Minsky) ແລະ ດີນ ເອັດມອນດ໌ (Dean Edmond) ນັກສຶກສາ ມະຫາວິທະຍາໄລ Princeton ໄດ້ຮ່ວມກັນສ້າງເຄືອຂ່າຍເສັ້ນປະສາດທຽມ (Neural Network) ໃຊ້ ຫຼອດສູນຍາກາດເຖິງ 3000 ຫຼອດ ຈຳລອງໜ່ວຍປະສາດ 40 ຫນ່ວຍ
- ອະລັນ ທົວລິງ (Alan Turing) ໄດ້ສະເໜີວິທີການທົດສອບວ່າໂປຣແກຣມສະຫຼາດ ຫຼືບໍ່, ວິທີນີ້ເອີ້ນວ່າ Turing Test, ເຄື່ອງຈັກຮຽນຮູ້, ການຮຽນຮູ້ແບບເສີມແຮງ ແລະ ອານກໍລຶທຶມດ້ານພັນທຸກຳ (Genetic Algorithm). ໃນຊ່ວງປີນີ້ໄດ້ມີການສ້າງແນວຄິດອັນເປັນພື້ນຖານຂອງການສ້າງຜົນງານປັນຍາປະດິດ.

<u>ການກຳເນີດ AI</u> (ຄ. ສ. 1956**)**

- ປັນຍາປະດິດກຳເນີດຢ່າງເຕັມຕົວທີ່ມະຫາວິທະຍາໄລ Princeton ໂດຍ ຈອຫ໌ນ ແມກຄາລ໌ທີ (John McCarthy) ໄດ້ຊວນ ມາລ໌ວິນ ມິນສກີ (Marvin Minsky), ຄໍດ ແຊນນອນ (Claude Shannon),, ນາ ທານຽນ ໂລເຊດເຕີລ໌ (Nathaniel Rochester) ແລະ ນັກວິໄຈຈາກສະຖາບັນອື່ນລວມ 10 ຄົນ ໃຫ້ ຊ່ວຍກັນເຮັດວິໄຈເລື່ອງທິດສະດີອັດຕະໂນມັດ (Automata Theory) ເຄືອຂ່າຍເສັ້ນປະສາດ ແລະ ສຶກ ສາເລື່ອງ "ຄວາມສະຫຼາດ: Intelligence"
- Allen Newell ແລະ Herbert Simon ໄດ້ພັດທະນາໂປຣແກຣມຫາເຫດຜົນ ແລະ ພິສູດທິດສະດີຕັກ ກະສາດ ຄືໂປຣແກຣມນັກທິດສະດີຕັກສາດ (Logic Theorist)
- ຜູ້ຕັ້ງຊື່ໃຫ້ ກັບວິທະຍາສາດສາຂາໃໝ່ນີ້ຄື John McCarthy ກ່ອນທີ່ຈະໄດ້ຊື່ວ່າ AI: Artificial Intelligence ນັ້ນມີອີກຊື່ໜຶ່ງທີ່ໜ້າຈະເໝາະຄື Computational Rationality
- AI ກ່ຽວຂ້ອງກັບວິທະຍາສາດດ້ານຈິດຕະວິທະຍາ, ກາຍະວິພາກ, ຄະນິດສາດ, ແຕ່ເປັນວິທະຍາສາດ ຍ່ອຍຂອງວິທະຍາສາດຄອມພິວເຕີ ເນື່ອງຈາກເນັ້ນການເຮັດໃຫ້ເຄື່ອງຈັກສາມາດເຮັດວຽກທີ່ມະນຸດເຮັດ ໄດ້ ຫຼື ເຮັດໄດ້ດີກວ່າ.

<u>ຄວາມຄາດຫວັງອັນຍິ່ງໃຫຍ່ທີ່ຕ້ອງການຈາກ AI (ຄ. ສ.</u> 1952-1969**<u>)</u>**

- AI ໃນຊ່ວງທຳອິດນີ້ຖືວ່າປະສົບຄວາມສຳເລັດ Newell ແລະ Simon ໄດ້ສ້າງອີກໂປຣແກຣມຄືໂປຣ ແກຣມແກ້ບັນຫາທົ່ວໄປ (GPS: General Problem Solver) ຈຳລອງວິທີການແກ້ບັນຫາໂດຍທົ່ວໄປ ຂອງມະນຸດ (Thinking Humanly) ໂດຍໄດ້ທົດລອງກັບບັນຫາການຕໍ່ຄຳ (Puzzle) ໃນຂອບເຂດ ຄວາມຍາກທີ່ກຳນົດ.
- McCarthy ໄດ້ສ້າງພາສາລະດັບສູງເພື່ອຂຽນໂປຣແກຣມດ້ານປັນຍາປະດິດໂດຍສະເພາະ ນັ້ນຄືມີຄວາມ ສາມາດໃນການຫາເຫດຜົນ ຈັດການກັບໂຈດບັນຫາທີ່ບໍ່ແມ່ນຕົວເລກ ພາສານີ້ຄື LISP

- McCullough ແລະ Pitts ສືບຕ່ອງກດ້ານເຄືອຂ່າຍເສັ້ນປະສາດທຽມ

<u>ສິ່ງທີ່ເຮັດໄດ້ ແລະ ສິ່ງທີ່ເຮັດບໍ່ໄດ້</u> (ຄ. ສ. 1966-1973)

- ໂປຣແກຣມສ່ວນຫຼາຍບໍ່ມີຄວາມຮູ້ໃນຂົງເຂດຄວາມຮູ້ທີ່ຈະເອົາມາໃຊ້ໃນການແກ້ໄຂບັນຫາ
- ຄວາມສາມາດໃນການໂຕ້ຕອບເຮັດໄດ້ຍາກ
- ແນວຄິດເລື່ອງເຄື່ອງຈັກການຍພັນ
- Minsky ແລະ Papert ໄດ້ຊີ້ໃຫ້ເຫັນເຖິງຂໍ້ຈຳກັດໃນການນຳໃຊ້ເຄືອຂ່າຍເສັ້ນປະສາດທຽມແບບໜຶ່ງຊັ້ນ, ເຊິ່ງສົ່ງຜົນໃຫ້ງານວິໄຈເລື່ອງນີ້ງຽບຫາຍໄປນັບ 10 ປີ, ເນື່ອງຈາກນັກວິໄຈເຊື່ອວ່າເຖິງທາງຕັນແລ້ວ.

<u>ລະບົບຖານຄວາມຮູ້</u> (ຄ. ສ.1969-1979)

- ລະບົບຖານຄວາມຮູ້ເປັນການແທນຂໍ້ມູນໃຫ້ໂປຣແກຣມທີ່ເຮັດວຽກດ້ານ AI ສາມາດນຳໄປໃຊ້ຫາເຫດຜົນ ຫຼືຫາຄຳຕອບໄດ້.
- ລະບົບສະໜັບສະໜູນການຕັດສິນໃຈເປັນການລວມຄວາມຮູ້ໃນຖານຄວາມຮູ້ກັບກົດເກນເພື່ອຊ່ວຍໃນການຕັດສິນໃຈ.
- ລະບົບຜູ້ຊ່ຽວຊານໄດ້ຖືກພັດທະນາຕໍ່ຈາກລະບົບສະໜັບສະໜູນການຕັດສິນໃຈ ໂດຍຮວບຮວມຄວາມຮູ້ໃນການແກ້ບັນຫາໜຶ່ງໆ ຈາກຜູ້ຊ່ຽວຊານໃນດ້ານນັ້ນໆ ແລະ ມີໂປຣແກຣມໃນການອ້າງເຫດຜົນ. ຜູ້ໃຊ້ປ້ອນຂໍ້ມູນລັກສະນະຂອງບັນຫາເຂົ້າລະບົບ, ແລ້ວໂປຣແກຣມໃນການອ້າງເຫດຜົນຈະຊອກຫາຄຳຕອບ ຫຼື ຄຳປຶກສາກັບຜູ້ໃຊ້.

<u>ອຸດສາຫະກຳ AI</u> (ຄ. ສ. 1980-ປັດຈຸບັນ)

ການໃຊ້ລະບົບຜູ້ຊ່ຽວຊານໃຫ້ຄຳປຶກສາໄດ້ເຂົ້າມາມີບົດບາດໃນວົງການອຸດສາຫະກຳ, ເຊັ່ນ: ລະບົບ R1 ຖືກໃຊ້ໃນບໍລິສັດ DEC ໃນການຊ່ວຍຫາການສັ່ງຊື້ລະບົບຄອມພິວເຕີໃໝ່ທີ່ເໝາະສົມ, ເຊິ່ງເຮັດໃຫ້ບໍລິສັດປະຫຍັດໄດ້ເຖິງປີລະ 40 ລ້ານດອນລາ. ໃນບໍລິສັດ DEC ໃຊ້ໂປຣແກຣມລະບົບຜູ້ຊ່ຽວຊານເຖິງ 40 ລະບົບ. ບໍລິສັດອື່ນອີກຫຼາຍບໍລິສັດຕ່າງກໍໃຫ້ຄວາມສົນໃຈ ແລະ ຍອມລົງທຶນມະຫາສານສ້າງໂປຣແກຣມປັນຍາປະດິດ, ແຕ່ບໍ່ປະສົບຄວາມສຳເລັດດັ່ງທີ່ຄາດໄວ້.

<u>ເຄືອຂ່າຍເສັ້ນປະສາດທຽມ -Neural Network</u> (ຄ. ສ.1986-ປັດຈຸບັນ)

- ຕັ້ງແຕ່ປີ 1970 ເຄືອຂ່າຍເສັ້ນປະສາດທຽມໄດ້ຮັບຄວາມສົນໃຈນ້ອຍຫຼາຍ ເນື່ອງຈາກນັກວິໄຈເຊື່ອວ່າບໍ່ສາມາດສ້າງໂປຣແກຣມທີ່ແກ້ບັນຫາໄດ້ແທ້, ແຕ່ເມື່ອນັກຟີຊິກຊື່ Hopfield ໄດ້ໃຊ້ວິທີການທາງສະຖິຕິກົນລະສາດວິເຄາະຄວາມຕ້ອງການໜ່ວຍຄວາມຈຳ ແລະ ຄຸນສົມບັດທີ່ເໝາະສົມທີ່ສຸດຂອງເຄືອຂ່າຍ ໂດຍເບິ່ງແຕ່ລະໜ່ວຍໃນເຄືອຂ່າຍເປັນອາຕອມ, ເຮັດໃຫ້ງານວິໄຈເຄືອຂ່າຍເສັ້ນປະສາດທຽມໄດ້ກັບມາອີກ
- ການຮຽນຮູ້ແບບແພ່ກະຈາຍກັບຄືນ (Backpropagation Learning) ໄດ້ຖືກສະເໜີໂດຍ Rumelhart, Hinton ແລະ Williams ສາມາດແກ້ບັນຫາໄດ້ທົ່ວໄປຫຼາຍຂຶ້ນ ແລະ ເຮັດໃຫ້ນັກວິໄຈກັບມາໃຫ້ຄວາມສົນໃຈເຄືອຂ່າຍເສັ້ນປະສາດທຽມອີກຄັ້ງ.

<u>ວິທະຍາສາດ AI</u> (ຄ. ສ. 1987-ປັດຈຸບັນ)

ປັນຍາປະດິດໄດ້ຮັບການຍອມຮັບເປັນວິທະຍາສາດສາຂາໜຶ່ງໃນປີ 1987 ເນື່ອງຈາກທີ່ຜ່ານມາມີການຄິດຄົ້ນວິທີການ, ທິດສະດີທີ່ເຮັດໃຫ້ການສ້າງເຄື່ອງຈັກທີ່ມີຄວາມສະຫຼາດ, ແກ້ບັນຫາໄດ້ຈິງ. ເຊັ່ນ: ຫຸ່ນຍົນ, ຄອມພິວເຕີວິຊັນ (Computer Vision), ການແທນຄວາມຮູ້, ການຮູ້ຈື່ຈຳສຽງ, ການຄົ້ນຫາຄວາມໝາຍທີ່ມີຢູ່ໃນຂໍ້ມູນຈຳນວນຫຼາຍ (Data Dining) ເປັນຕົ້ນ.

<u>ຕົວແທນປັນຍາປະດິດ</u> (ຄ. ສ.1995-ປັດຈຸບັນ)

ໃນການແກ້ບັນຫາຂອງໂປຣແກຣມປັນຍາປະດິດນັ້ນ ຖ້າເປັນບັນຫາຊັບຊ້ອນ ແລະ ບັນຫານັ້ນມີການຄອຍຖ້າຕິດຕໍ່ ຫຼື ຕິດຕາມເບິ່ງການປ່ຽນແປງຕ່າງໆຂອງສິ່ງແວດລ້ອມ ມັກຈະແບ່ງວຽກອອກເປັນວຽກຍ່ອຍ, ແລ້ວມີຕົວແທນປັນຍາ (Intelligent Agent) ເຮັດວຽກໃນສ່ວນຍ່ອຍ. ຕົວຢ່າງທີ່ເຫັນໄດ້ແຈ້ງທີ່ສຸດຄືຕົວແທນປັນຍາເທິງອິນເຕີເນັດ, ເຊິ່ງເຮົາມັກຈະເອີ້ນວ່າ bot (ບອດ) ເຊັ່ນ: ບອດຂອງໂປຣແກຣມຄົ້ນຫາຂໍ້ມູນ (Search Engine).

1.3 ການປະຍຸກໃຊ້ AI ໃນປັດຈຸບັນ

ປັດຈຸບັນງານວິໄຈທາງດ້ານປັນຍາປະດິດ ໄດ້ມີການນຳໄປປະຍຸກໃຊ້ໃນການແກ້ໄຂບັນຫາຕ່າງໆຢ່າງກວ້າງຂວາງ ໃນຫຼາຍໆສາຂາ, ແຕ່ວຽກສ່ວນໃຫຍ່ຈະເນັ້ນໜັກໄປໃນຮູບແບບທີ່ໃຊ້ເຫດຜົນເປັນຫຼັກ. ເນື່ອງຈາກສາຂາທີ່ມີການນຳປັນຍາປະດິດໄປປະຍຸກໃຊ້ແກ້ບັນຫາ ບໍ່ຈຳເປັນຕ້ອງອາໄສອາລົມ ຫຼື ຄວາມຮູ້ສຶກຂອງມະນຸດ. ຕໍ່ໄປນີ້ຈະກ່າວເຖິງສາຂາເຊິ່ງເປັນທີ່ຮູ້ຈັກກັນຢ່າງແພ່ຫຼາຍໄດ້ແກ່:

- ການປະມວນຜົນພາສາທຳມະຊາດ (Natural Language Processing)

ເປັນການສຶກສາທີ່ເນັ້ນໃຫ້ຄອມພິວເຕີສາມາດສື່ສານກັບຜູ້ໃຊ້ໄດ້ໂດຍໃຊ້ພາສາທຳມະຊາດ ງານວິໄຈສາຂານີ້ຕ້ອງອາໄສຄວາມຮູ້ທາງດ້ານພາສາສາດ ເພື່ອສຶກສາເຖິງວິທີການປະມວນຜົນລວມທັງການຮັບຮູ້, ການເຂົ້າໃຈ ແລະ ການນຳໃຊ້ພາສາທຳມະຊາດ ເພື່ອໃຫ້ຄອມພິວເຕີສາມາດເຂົ້າໃຈພາສາມະນຸດໄດ້. ຕົວຢ່າງງານວິໄຈດ້ານນີ້ໄດ້ແກ່: ເຄື່ອງແປພາສາ, ລະບົບຄົ້ນຄືນໂດຍໃຊ້ພາສາທຳມະຊາດ, ລະບົບສອບຖາມທາງໂທລະສັບອັດຕະໂນມັດ.

- ວິທະຍາການຫຸ່ນຍົນ (Robotics)

ໂດຍທົ່ວໄປຫຸ່ນຍົນໝາຍເຖິງເຄື່ອງຈັກທີ່ສາມາດເຮັດວຽກໃນທາງກາຍະພາບຕ່າງໆໄດ້, ເຊິ່ງຫຸ່ນຍົນເຫຼົ່ານີ້ຈະປະກອບດ້ວຍ Effectors ເຊັ່ນ: ຂາວົງລໍ້, ຂໍ້ຕໍ່, ຕົວຢຶດຈັບ ເຊິ່ງສ່ວນທີ່ເຮັດໜ້າທີ່ສົ່ງແຮງອອກໄປຫາສະພາບແວດລ້ອມ ແລະ Sensors ທີ່ເປັນຕົວຮັບຮູ້ສະພາບແວດລ້ອມ. ງານວິໄຈໃນສາຂານີ້ຈຳເປັນຕ້ອງອາໄສຄວາມຮູ້ທາງກົນຈັກ ເພື່ອເຮັດໃຫ້ຫຸ່ນຍົນສາມາດເຄື່ອນໄຫວໄດ້ຕາມທີ່ໂປຣແກຣມຄວບຄຸມສັ່ງການ. ປັດຈຸບັນຫຸ່ນຍົນອາດຈະແບ່ງໄດ້ເປັນ 3 ກຸ່ມຫຼັກໆ, ກຸ່ມທຳອິດໄດ້ແກ່: ແຂນກົນ ຫຼື Manipulators ເຊິ່ງສ່ວນໃຫຍ່ມັກຈະໃຊ້ໃນວົງການອຸດສາຫະກຳ, ເຊັ່ນ: ວຽກຍົກເຄື່ອງ, ວຽກປະກອບຊິ້ນສ່ວນຕ່າງໆ ຫຼື ລວມທັງໃນວົງການແພດກໍມີແຂນກົນທີ່ຊ່ວຍໃນການຜ່າຕັດ. ກຸ່ມທີ່ສອງເອີ້ນວ່າ ຫຸ່ນຍົນເຄື່ອນທີ່ ຫຼື Mobile Robot ເຊິ່ງສາມາດເຄື່ອນໄຫວໃຫ້ເຂົ້າກັບສິ່ງແວດລ້ອມໄດ້, ຕົວຢ່າງເຊັ່ນ: ຫຸ່ນຍົນທີ່

ສາມາດເຄື່ອນທີ່ໄດ້ເທິງພື້ນຜິວ, ໃຕ້ນ້ຳ ຫຼື ໃນອາກາດ ແລະ ກຸ່ມສຸດທ້າຍແມ່ນຫຸ່ນຍົນແບບປະສົມ ຫຼື hybrid ທີ່ລວມເອົາຄວາມສາມາດຂອງຫຸ່ນຍົນສອງກຸ່ມທຳອິດເຂົ້າກັນ, ໂດຍລວມທັງຫຸ່ນຍົນທີ່ເອີ້ນວ່າ Humanoid ເຊິ່ງເປັນຫຸ່ນຍົນທີ່ອອກແບບໃຫ້ມີຮູບຮ່າງຄ້າຍຄືມະນຸດ ຕົວຢ່າງທີ່ມີຊື່ສຽງຂອງຫຸ່ນຍົນແບບນີ້ ໄດ້ແກ່ ASIMO ຂອງບໍລິສັດຮອນດ້າ.

- ການພິສູດທິດສະດີ (Theorem Proving)

ເປັນການພັດທະນາໂປຣແກຣມຄອມພິວເຕີເພື່ອໃຊ້ພິສູດທິດສະດີໂດຍອາໄສກົດເກນທາງຕັກສາດ (Predicate Logic) ເຊິ່ງສາມາດນຳໄປໃຊ້ການພິສູດທິດສະດີບົດທາງຄະນິດສາດ ຫຼື ນຳໄປໃຊ້ໃນການກວດສອບການອອກແບບວົງຈອນ.

- ລະບົບຜູ້ຊ່ຽວຊານການ (Expert Systems)

ເປັນງານວິໄຈທີ່ເນັ້ນສ້າງລະບົບຜູ້ຊ່ຽວຊານການເພື່ອໃຫ້ຄຳປຶກສາ ແລະ ຄຳຕອບກ່ຽວກັບບັນຫາຕ່າງໆ, ໂດຍລະບົບຈະເກັບລວບລວມຄວາມຮູ້ ແລະ ຂໍ້ມູນຈາກປະສົບການຂອງຜູ້ຊ່ຽວຊານການ ແລະ ສາມາດນຳຄວາມຮູ້ນັ້ນມາວິເຄາະເພື່ອຫາຄຳຕອບຂອງບັນຫາ. ງານວິໄຈສາຂານີ້ຈຳເປັນຕ້ອງອາໄສຄວາມຮູ້ພື້ນຖານຫຼາຍໆ ເຊັ່ນ: ການແທນຄວາມຮູ້, ການຄິດໃຫ້ເຫດຜົນ ແລະ ການຮຽນຮູ້ດ້ວຍເຄື່ອງ. ຕົວຢ່າງງານວິໄຈດ້ານນີ້ທີ່ຮູ້ກັນຢ່າງແພ່ຫຼາຍໄດ້ແກ່: ລະບົບ MYCIN ທີ່ໃຊ້ວິເຄາະພະຍາດທີ່ເກີດຈາກເຊື້ອແບັດທີເລຍ, ລະບົບ DENDRAL ທີ່ສາມາດວິເຄາະສະເປັກໂຕກຣາມ (Spectrogram) ຂອງມວນຕ່າງໆ.

- ການຂຽນໂປຣແກຣມອັດຕະໂນມັດ (Automatic Programming)

ເປັນການເຮັດໃຫ້ຄອມພິວເຕີສ້າງໂປຣແກຣມໄດ້ອັດຕະໂນມັດ ໂດຍສາມາດແປພາສາຊັ້ນສູງ ຫຼື ພາສາທີ່ໃກ້ຄຽງກັບພາສາທຳມະຊາດ (ແທນການໃຊ້ພາສາໂປຣແກຣມທົ່ວໄປ) ໃຫ້ເປັນໂປຣແກຣມໃນພາສາເຄື່ອງໄດ້, ເຊິ່ງເຮົາອາດເຫັນໄດ້ວ່າການຂຽນໂປຣແກຣມອັດຕະໂນມັດນີ້ກໍຄື Compiler ຊະນິດພິເສດຊະນິດໜຶ່ງໄດ້.

- ບັນຫາການຈັດຕາຕະລາງ (Scheduling Problems)

ເປັນການນຳເອົາປັນຍາປະດິດໄປໃຊ້ໃນການແກ້ບັນຫາການກຳນົດຕາຕະລາງເວລາ ຫຼື ການເລືອກ Combination ໃຫ້ເໝາະສົມທີ່ສຸດ, ເຊັ່ນ: ການຈັດຕາລາງການເດີນທາງຂອງພະນັກງານຂາຍເຄື່ອງໂດຍເສຍຄ່າໃຊ້ຈ່າຍນ້ອຍທີ່ສຸດ, ການຈັດຕາລາງການຂຶ້ນລົງຂອງຍົນໃຫ້ເກີດປະໂຫຍດສູງສຸດ, ການຈັດຕາລາງການຮຽນການສອນໃຫ້ເກີດປະສິດທິພາບສູງສຸດ ເປັນຕົ້ນ.

- ບັນຫາດ້ານປະສາດສຳຜັດ (Perception Problems)

ເປັນການປະຍຸກໃຊ້ເພື່ອໄປແກ້ບັນຫາການຮັບຮູ້ຕ່າງໆ ເຊັ່ນ: ການເບິ່ງ-ເຫັນ, ການສຳຜັດ, ການຟັງ ຕົວຢ່າງເຊັ່ນ: ການສ້າງຫຸ່ນຍົນທີ່ສາມາດຍ່າງຫລົບຫລີກສິ່ງກີດຂວາງໄດ້.

❖ ສິ່ງທີ່ປັນຍາປະດິດຍັງເຮັດບໍ່ທັນໄດ້ເທື່ອ

- ເຂົ້າໃຈພາສາທຳມະຊາດໄດ້ຢ່າງດີ ເຊັ່ນ: ອ່ານ ແລະ ເຂົ້າໃຈຫົວຂໍ້ຕ່າງໆໃນໜັງສືພິມ

- ການທ່ອງໂລກອິນເຕີເນັດ (Surf the web)
- ຕີຄວາມໝາຍຂອງຮູບພາບຕາມໃຈ
- ຮຽນພາສາທຳມະຊາດ
- ສ້າງແຜນໃນຂອບເຂດເວລາຈິງໄດ້ຢ່າງຄ່ອງແຄ້ວ
- ສະແດງໃຫ້ເຫັນລັກສະນະ ແລະ ຄວາມສະຫຼາດທີ່ແທ້ຈິງ

1.4 ປະເພດຂອງປັນຍາປະດິດ (AI Category)

ໃນຕາຕະລາງລຸ່ມນີ້ແມ່ນຕົວຢ່າງຂອງປັນຍາປະດິດ, ເຊິ່ງສ່ວນຫຼາຍແມ່ນປັນຍາປະດິດຍຸກໃໝ່.

ປະເພດ	ຄຳອະທິບາຍ
Affective Computing	ການສຶກສາແລະການພັດທະນາຂອງລະບົບ ແລະອຸປະກອນ ທີ່ສາມາດຮັບຮູ້, ຕີຄວາມຫມາຍ, ດຳເນີນການ, ແລະ ການຈຳລອງ ຄວາມເປັນມະນຸດ.
Artificial Immune Systems	ລະບົບການຮຽນຮູ້ແບບອັດສະລິຍະທີ່ອີງໃສ່ຫຼັກການ ແລະ ຂະບວນການ ເຊິ່ງມີຢູ່ໃນລະບົບພູມຕ້ານທານຂອງສັດລ້ຽງລູກດ້ວຍນົມ
Chatterbot	ປະເພດຂອງຕົວແທນການສົນທະນາ ຫຼື ໂປຣແກຣມຄອມພິວເຕີທີ່ອອກແບບມາເພື່ອກະຕຸ້ນການສົນທະນາອັດສະລິຍະກັບຜູ້ໃຊ້ຄົນໜຶ່ງຫຼືຫຼາຍຄົນຜ່ານຂໍ້ຄວາມຫຼືສຽງ
Cognitive Architecture	ທິດສະດີກ່ຽວກັບໂຄງສ້າງຂອງຈິດໃຈມະນຸດ. ໜຶ່ງໃນເປົ້າໝາຍຫຼັກຄື ການໂຮມແມວຄິດຈາກຈິດຕະສາດ, ຄວາມຮູ້ສຶກ, ຄວາມເຂົ້າໃຈ ໃສ່ໃນແບບຈຳລອງຄອມພິວເຕີທີ່ຄວບຄຸມ
Computer Vision	ສາຂາວິທະຍາສາດແບບກວ້າງທີ່ກ່ຽວຂ້ອງກັບວິທີການທີ່ຄອມພິວເຕີສາມາດຮັບຄວາມເຂົ້າໃຈລະດັບສູງຈາກຮູບພາບ ແລະ ວີດີໂອ.
Evolutionary Computing	ການໃຊ້ຂັ້ນຕອນວິທີທີ່ມີວິວັດທະນາການຕາມຫຼັກການຂອງ ດາວິນຽນ (Darwinian)ເຊິ່ງເປັນທີ່ມາຂອງຊື່ນີ້. ຂັ້ນຕອນວິທີເຫຼົ່ານີ້ຢູ່ໃນຕະກຸນຂອງຕົວແກ້ບັນຫາແບບການທົດລອງແລະຂໍ້ຜິດພາດ (trial-and-error) ແລະ ໃຊ້ວິທີ ເມຕ້າຮິວຣິດສະຕິກ (metaheuristics) ຫຼື stochastic global ເພື່ອກຳນົດການແກ້ໄຂແບບຕ່າງໆ
Gaming AI	AI ໃຊ້ໃນເກມເພື່ອສ້າງພຶດຕິກຳສະຫຼາດ, ໂດຍສ່ວນຫຼາຍຢູ່ໃນຕົວທີ່ບໍ່

	ແມ່ນຜູ້ຫຼີ້ນ (non-player characters :NPCs)ເຊິ່ງມັກຈະຈຳລອງຄວາມສະຫຼາດຄືມນຸດ.
Human-Computer-Interface (HCI)	HCI ຄົ້ນຄ້ວາການອອກແບບ ແລະ ການນຳໃຊ້ເຕັກໂນໂລຊີຄອມພິວເຕີ, ເນັ້ນໃສ່ການເຊື່ອມຕໍ່ລະຫວ່າງຄົນ (ຜູ້ໃຊ້) ແລະ ຄອມພິວເຕີ.
Intelligent Soft Sssistant or Intelligent Personal Assistant (IPA)	ຕົວແທນຊອບແວທີ່ສາມາດປະຕິບັດວຽກຫຼືບໍລິການສຳລັບບຸກຄົນໃດໜຶ່ງ. ໜ້າວຽກ ຫຼື ການບໍລິການ ເຫຼົ່ານີ້ຕາມປົກກະຕິແມ່ນອີງໃສ່ການປ້ອນຂໍ້ມູນຂອງຜູ້ໃຊ້, ການຮັບຮູ້ສະຖານທີ່ ແລະ ຄວາມສາມາດໃນການເຂົ້າເຖິງຂໍ້ມູນຈາກແຫຼ່ງຂໍ້ມູນອອນໄລນ໌. ຕົວຢ່າງຂອງຕົວແທນດັ່ງກ່າວແມ່ນ Apple's Siri, Amazon's Alexa, Amazon's Evi, Google's Home, Microsoft Cortana, ແຫຼ່ງເປີດ Lucida, Braina (ແອບພິເຄຊັນທີ່ພັດທະນາໂດຍ Brainasoft ສຳລັບ Microsoft Windows), S Voice ຂອງ Samsung ແລະ LG G3 Voice Mate.
Knowledge Engineering	ຫມາຍເຖິງທັງໝົດຂອງ ດ້ານວິຊາການ, ວິທະຍາສາດແລະສັງຄົມທີ່ກ່ຽວຂ້ອງໃນການສ້າງ, ການຮັກສາແລະ ນຳໃຊ້ລະບົບຄວາມຮູ້.
Knowledge Representation	ເຊິ່ງເປັນຕົວແທນຂອງຂໍ້ມູນກ່ຽວກັບໂລກໃນຮູບແບບທີ່ລະບົບຄອມພິວເຕີສາມາດນຳໃຊ້ເພື່ອແກ້ໄຂວຽກງານທີ່ສັບສົນເຊັ່ນ: ການວິນິດໄສສະພາບທາງການແພດຫຼືມີການສົນທະນາໃນພາສາທຳມະຊາດ.
Logic Programming	ປະເພດຂອງການຂຽນໂປຣແກຣມສ່ວນຫຼາຍແມ່ນອີງໃສ່ຕາມຫຼັກຕັກກະສາດ. ໂປຣແກຣມໃດໆທີ່ຂຽນໃນພາສາໂປຼແກຼມແບບບຕັກກະສາດເປັນຊຸດຂອງປະໂຫຍກໃນແບບຕັກກາສາດ, ສະແດງຂໍ້ເທັດຈິງແລະກົດກ່ຽວກັບບາງຂອບເຂດບັນຫາ. ຕະກູນຫຼັກໆຂອງພາສາໂປຼແກຼມແບບບຕັກກະສາດມີ ພາສາ PROLOG, Answer Set Programming (ASP) ແລະ Datalog.
Machine Learning	ML ໃນເນື້ອໃນຂອງ AI ໃຫ້ຄອມພິວເຕີສາມາດຮຽນຮູ້ໄດ້ເອງໂດຍບໍ່ມີການຕັ້ງໂປຣແກຣມໄວ້ຢ່າງຊັດ. ການຮຽນຮູ້ແບບຕື້ນແລະແບບເລິກເປັນສາຂາຍ່ອຍທີ່ສຳຄັນ.
Multi-Agent System	MAS ຄືລະບົບທີ່ເຮັດໃຫ້ Agent ຫຼາຍໆເຄື່ອງເຮັດວຽກຮ່ວມກັນ, ເຊິ່ງ Agent ເຫຼົ່ານີ້ແຕ່ລະໂຕສາມາດເຮັດວຽກໄດ້ຢ່າງອິດສະຫຼະ ແລະທີ່ສຳຄັນແຕ່ລະໂຕາມາດຕັດິນໃຈໄດ້ດ້ວຍຕົວເອງ
Robotics	Robotics ເປັນສາຂາທີ່ໃຫຍ່ທາງດ້ານວິສະວະກຳ ແລະ ວິທຍາສາດທີ່

	ປະກອບມີ ວິສະວະກຳຈັກກົນ, ວິສະວະກຳໄຟຟ້າ, ວິທະຍາສາດຄອມພິວເຕີ, ປັນຍາປະດິດ ແລະ ອື່ນໆ
Robots	ຫຸ່ນຍົນເປັນເຄື່ອງຈັກ,ໂດຍສະເພາະມີການຕັ້ງໂປຣແກຣມດ້ວຍຄອມພິວເຕີ. ເຊິ່ງສາມາດປະຕິບັດວຽກທີ່ຊັບຊ້ອນຂອງການດຳເນີນງານໄດ້ດ້ວຍຕົວເອງ
Rule Engines or Systems	Rule-based systems ຖືກໃຊ້ເພື່ອຈັດເກັບແລະຈັກການຄວາມຮູ້ເພື່ອຕີຄວາມໝາຍຂໍ້ມູນໃນທາງທີ່ເປັນປະໂຫຍດ.
Turing Test	Turing Test ແມ່ນການທົດສອບ, ທີ່ພັດທະນາໂດຍ Alan Turing ໃນ 1950, ຄວາມສາມາດຂອງເຄື່ອງທີ່ຈະສະແດງພຶດຕິກຳທີ່ສະຫຼາດທຽບເທົ່າກັບ, ຫຼືບໍ່ສາມາດແຕກຕ່າງຈາກ, ນັ້ນແມ່ນມະນຸດ.

1.5 ປັນຍາປະດິດ ແລະ ບິກດາຕ້າ (AI and Big data)

ເຮົາອາດຈະເຄີຍໄດ້ຍິນຄຳວ່າ *Big Data*, ແຕ່ວ່າຄົນສ່ວນຫຼາຍອາດຈະບໍ່ຮູ້ວ່າມັນແມ່ນຫຍັງ ແລະ ມີຜົນກະທົບຍັງຕໍ່ກັບສັງຄົມຍຸກໃໝ່. Big Data ມີຫຼາຍນິຍາມ, ຄືກັນກັບປັນຍາປະດິດທີ່ມີຫຼາຍນິຍາມຄືກັນ. ນິຍາມຕໍ່ໄປນີ້ເປັນນິຍາມແບບທົ່ວໆໄປ: ຊຸດຂໍ້ມູນໜຶ່ງທີ່ປະກອບມີ 3ອົງປະກອບຄື: ຂໍ້ມູນຂະໜາດໃຫຍ່, ຄວາມໄວ ແລະ ຄວາມຫຼາກຫຼາຍ (*a data collection characterized by huge volumes, rapid velocity, and great variety).*

ຂໍ້ມູນຂະໜາດໃຫຍ່ (Huge Volumes) ນີ້ສາມາດວັດແທກໄດ້ດ້ວຍສັນຍາລັກເປຕາໄບ (Petabytes :PB), ເຊິ່ງ 1 PB ຈະເທົ່າກັບ 1 ລ້ານກິກກະໄບ (Gigabytes) ແລະ ນັ້ນໝາຍເຖິງຂໍ້ມູນຂະໜາດໃຫຍ່ ຄວາມໄວ (Rapid Velocity) ໃນນີ້ໝາຍເຖິງຄວາມໄວຂອງຂໍ້ມູນທີ່ຖືກຜະລິດ ແລະ ສ້າງຂຶ້ນ, ຕົວຢ່າງ ຄວາມໄວຂອງຂໍ້ມູນ ໃນ Facebook ໃນແຕ່ລະວິນາທີ ໄດ້ມີເນື້ອໃນເກີດຂຶ້ນໃໝ່ເລື້ອຍໆທີ່ຖືກສ້າງຈາກຜູ້ໃຊ້ອອນໄລຫຼາຍລ້ານຄົນພ້ອມກັນ. ຄວາມຫຼາກຫຼາຍຂະໜາດໃຫຍ່ (Great Variety) ໃນທີ່ນີ້ໝາຍເຖິງ ຊະນິດຂໍ້ມູນແບບຕ່າງໆ ທີ່ເຂົ້າສູ່ການໄຫຼຂອງກະແສຂໍ້ມູນຂະໜາດມະໂຫລານ, ລວມມີ ຮູບ, ພາບເຄື່ອນໄຫວ ແລະ ໂຕໜັງສື. ໜຶ່ງຮູບທີ່ຖືກອັບໂລດຢູ່ໃນ Facebook ສະເລ່ຍປະມານ 4 ຫາ 5 ເມັກກາໄບ, ພວກເຮົາລອງຄິດໄລ່ເບິ່ງວ່າຮູບຈຳນວນຫຼາຍລ້ານຮູບທີ່ຖືກອັບໂລດຂຶ້ນຕະຫຼອດເວລາ, ເຊິ່ງສິ່ງເຫຼົ່ານີ້ແມ່ນ ລັກສະນະ ຂອງ Big Data. AI ມີຜົນຫຍັງກັບ Big Data? ຄຳຕອບກໍ່ແມ່ນ ລະບົບການຮຽນຮູ້ AI ເມື່ອໃຊ້ກັບຊຸດຂໍ້ມູນຂະໜາດໃຫຍ່ແລ້ວຊ່ວຍໃຫ້ຜູ້ໃຊ້ສາມາດດຶງຂໍ້ມູນທີ່ເປັນປະໂຫຍດຈາກການນຳເຂົ້າຂໍ້ມູນຂະໜາດໃຫຍ່ ແລະ ຂໍ້ມູນທີ່ບໍ່ຄົບຖ້ວນ. ລະບົບຄອມພິວເຕີສາມາດຈັດການກັບຂໍ້ມູນຂະໜາດໃຫຍ່ໄດ້

ປະກອບມີໜ່ວຍປມອນຜົນຫຼາຍພັນໂຕທີ່ເຮັດວຽກຂະໜານກັນເພື່ອເພີ່ມຄວາມໄວຂອງຂະບວນ, ການຫຼຸດ ຂໍ້ມູນທີ່ຖືກເອີ້ນວ່າ MapReduce. ຄອມພິວເຕີ IBM's Watson ເປັນລະບົບຕົວຢ່າງທີ່ສຳຄັນ, ມີການໃຊ້ ລະບົບຜູ້ຊ່ຽວຊານທາງການແພດໂດຍໃຊ້ Rule-Based Engine ແລະ ປະມວນຜົນຂໍ້ມູນທາງການແພດ ຫຼາຍພັນ Record ເກືອບວ່າຮອດລ້ານ, ຜົນທີ່ໄດ້ຄືລະບົບຄອມພິວເຕີທີ່ຊ່ວຍແພດບົ່ງມະຕິພະຍາດ ແລະ ພະຍາດກ່ຽວຂ້ອງທີ່ບໍ່ສະແດງອາການ ຫຼື ມີຄວາມກ່ຽວຂ້ອງກັບພະຍາດທີ່ຮູ້.

ບົດທີ 2 ການແກ້ປັນຫາດ້ວຍເຕັກນິກການຄົ້ນຫາ (Problem Solving using Search)

2.1. ຂອບເຂດຂອງປັນຫາ (State Space)

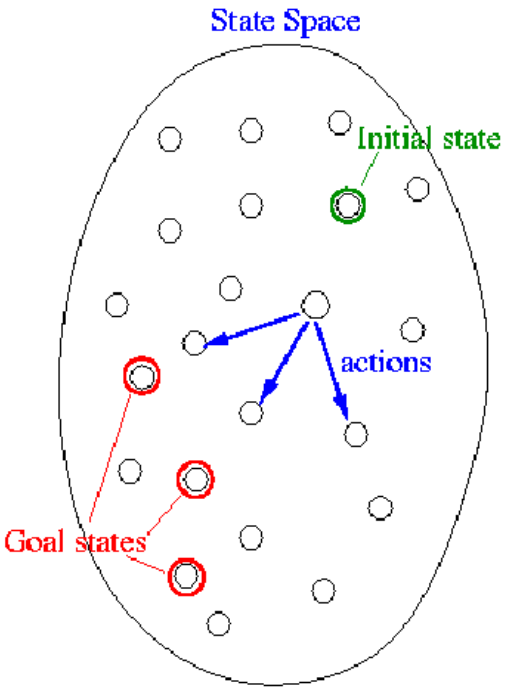
ປັນຫາພື້ນຖານໃນການພັດທະນາລະບົບປັນຍາປະດິດກໍຄືຜູ້ພັດທະນາຈະຕ້ອງເຂົ້າໃຈໃນເຕັກນິກຂອງການສ້າງໂປຣແກຣມ ເພື່ອຈະເຮັດໃຫ້ຄອມພິວເຕີສາມາດປະມວນຜົນໄດ້, ນັກວິໄຈທາງດ້ານປັນຍາປະດິດໃນຍຸກທຳອິດພະຍາຍາມຫາຄຳຕອບນີ້ ແລະ ຄົ້ນພົບວ່າລັກສະນະຢ່າງໜຶ່ງຂອງການນຳໃຊ້ໂປຣແກຣມຄອມພິວເຕີສາມາດເຮັດສິ່ງດັ່ງກ່າວໄດ້ກໍຄື ລະບົບປັນຍາປະດິດຈະຕ້ອງມີຄວາມຮູ້ ແລະ ມີລະບົບການຫາຂໍ້ສະຫຼຸບຄວາມຮູ້ນັ້ນໄດ້, ລະບົບນີ້ກໍຈະສາມາດແກ້ປັນຫາບາງຢ່າງໄດ້ຄືກັນກັບມະນຸດ. ແຕ່ສິ່ງທີ່ເຮັດໃຫ້ຄອມພິວເຕີມີຄວາມຮູ້ ແລະ ຄວາມສາມາດຫາຂໍ້ສະຫຼຸບໄດ້ບໍ່ແມ່ນເລື່ອງທີ່ງ່າຍ ຍ້ອນຄວາມຮູ້ເປັນຖານຂໍ້ມູນທີ່ມີຂະໜາດໃຫຍ່ຫຼາຍ ແລະ ການຫາຂໍ້ສະຫລຸບກໍເປັນຂະບວນການແກ້ປັນຫາມີຄວາມຊັບຊ້ອນຫຼາຍເຊັ່ນກັນ.

ເມື່ອປີ 1976 ເນວິວ ແລະ ໄຊມອນ (Newell and Simon, 1976)ໄດ້ສະເໜີແນວຄິດກ່ຽວກັບການເຮັດວຽກທີ່ເປັນອັດສະລິຍະວ່າສາມາດເກີດຂຶ້ນໄດ້ດ້ວຍຂະບວນການດັ່ງນີ້ :

- ການໃຊ້ຮູບແບບຂອງສັນຍາລັກ (Symbolic Pattern) ແທນຄ່າອົງປະກອບທີ່ສຳຄັນຂອງປັນຫາໃນຂອບເຂດຂອງຄວາມຮູ້ນັ້ນ.
- ການປະມວນຜົນຮູບແບບຂອງສັນຍະລັກເຫຼົ່ານີ້ເພື່ອຫາຄຳຕອບທີ່ເປັນໄປໄດ້ (Possible Solution) ທັງໝົດອອກມາ.
- ຄົ້ນຫາເພື່ອເລືອກຄຳຕອບຈາກຄຳຕອບທີ່ເປັນໄປໄດ້ທັງໝົດທີ່ຖືກສ້າງຂຶ້ນມາໃນຂັ້ນຕອນກ່ອນໜ້ານັ້ນ.

ຂໍ້ສະຫຼຸບຂອງເນວິວ ແລະ ໄຊມອນ ເຮັດໃຫ້ເກີດການແກ້ປັນຫາແບບ ຂອບເຂດສະຖານະ (State Space) ເຊິ່ງເປັນວິທີການແກ້ປັນຫາແບບໜຶ່ງທີ່ກຳນົດວ່າປັນຫາມີຈຸດເລີ່ມຕົ້ນ ແລະ ເປົ້າໝາຍ ຂອງການແກ້ປັນຫາ, ເຊິ່ງເອີ້ນວ່າສະຖານະຈຸດເລີ່ມຕົ້ນ ຫຼື ສະຖານະເລີ່ມຕົ້ນ (Initial State) ເປັນສະຖານະທີ່ກຳນົດຈຸດເລີ່ມຕົ້ນຂອງການແກ້ປັນຫາ ຫຼື ໂຈດຂອງປັນຫາ, ສຳລັບເປົ້າໝາຍ ຫຼື ສະຖານະເປົ້າໝາຍ (Gold State) ຄືສິ່ງທີ່ເຮົາຄາດຫວັງຈະໄດ້ຈາກການແກ້ປັນຫານີ້ຄຳຕອບ. ສະຖານະທັງສອງນີ້ຈະຢູ່ຫ່າງຈາກັນແລະ ສິ່ງທີ່ເຮົາຈະຫາທາງໄປຈາກສະຖານະເລີ່ມຕົ້ນມາສະຖານະເປົ້າໝາຍໄດ້ຈະຕ້ອງຜ່ານສະຖານະຕ່າງໆທີ່ຢູ່ລະຫວ່າງກາງຈຳນວນຫຼາຍ ແລະ ສະຖານະເຫຼົ່ານີ້ສາມາດທຽບໄດ້ກັບຄຳຕອບຍ່ອຍໆທີ່ເຮົາໄດ້ຮັບຈາກການແກ້ປັນຫາກ່ອນທີ່ຈະໄດ້ຄຳຕອບສຸດທ້າຍ, ສະຖານະເຫຼົ່ານີ້ລວມກັນເອີ້ນວ່າ ພື້ນທີ່ປັນຫາ (Problem Space). ໃນພື້ນທີ່ປັນຫາຈະມີບາງສະຖານະເທົ່ານັ້ນທີ່ສາມາດເປັນທາງຜ່ານໄປສູ່ເປົ້າໝາຍ ຫຼື ຄຳຕອບສຸດທ້າຍໄດ້, ແຕ່ບາງສະຖານະຈະບໍ່ສາມາດພາໄປສູ່ເປົ້າໝາຍ ຫຼື ຄຳຕອບສຸດທ້າຍໄດ້, ຄວາມຫ່າງຂອງສະຖານະເລີ່ມຕົ້ນ ແລະ ສະຖານະເປົ້າໝາຍນີ້ກຳນົດໂດຍຈຳນວນສະຖານະທີ່ຢູ່ລະຫວ່າງກາງ ຖ້າສະຖານະເລີ່ມຕົ້ນ ແລະ ສະຖານະເປົ້າໝາຍຫ່າງກັນຫຼາຍ ໝາຍຄວາມວ່າຂະໜາດຂອງພື້ນທີ່ປັນຫານີ້ກໍ່ຈະໃຫຍ່ ຖ້າຄວາມຫ່າງນ້ອຍຂະໜາດພື້ນທີ່ປັນຫາກໍ່ຈະນ້ອຍ. ເມື່ອເລີ່ມແກ້ປັນຫາການເຮັດວຽກຂອງລະບົບຈະຢູ່ທີ່ສະຖານະເລີ່ມຕົ້ນ ຈາກນັ້ນຈະຕ້ອງພະຍາຍາມຫາທາງໄປຈາກສະຖານະ

ເລີ່ມຕົ້ນນີ້ຜ່ານສະຖານະຕ່າງໆທີ່ຢູ່ລະຫວ່າງກາງ ເພື່ອໄປສູ່ສະຖານະເປົ້າໝາຍໃຫ້ໄດ້ ເມື່ອສາມາດໄປຫາສະຖານະເປົ້າໝາຍໄດ້ ການແກ້ປັນຫາກໍ່ຈະສິ້ນນສຸດຄືດັ່ງສະແດງໃນຮູບທີ 2.1



ຮູບທີ 2.1 State Space

ປຽບທຽບການແກ້ປັນຫາຂອງພື້ນທີ່ປັນຫາເປັນຮູບປະທຳກໍແມ່ນການຫາທາງໄປເຂົ້າວົງກົດສະຖານະເລີ່ມຕົ້ນຄືທາງເຂົ້າ ແລະ ສະຖານະເປົ້າໝາຍຄືທາງອອກ, ສຳລັບເສັ້ນທາງຕ່າງໆໃນເຂົ້າວົງກົດແມ່ນສະຖານະຕ່າງໆ, ໃນກໍລະນີນີ້ເຮົາຈະເຫັນໄດ້ວ່າມີເສັ້ນທາງບາງເສັ້ນທາງເທົ່ານັ້ນທີ່ພາໄປສູ່ທາງອອກ ແລະ ເສັ້ນທາງສ່ວນຫຼາຍຈະເປັນເສັ້ນທາງທີ່ພາໃຫ້ຫຼົງ, ເຊັ່ນດຽວກັນກັບພື້ນທີ່ປັນຫາ, ເສັ້ນທາງພື້ນທີ່ປັນຫາຈະມີຫຼາຍ ແຕ່ມີພຽງບາງເສັ້ນທາງເທົ່ານັ້ນທີ່ພາໄປສູ່ຄຳຕອບໄດ້.

ໃນການແກ້ປັນຫາແບບຂອບເຂດສະຖານະ (State Space) ເປັນການແທນປັນຫາທັງໝົດອອກມາໃນຮູບແບບຂອງ ກຣາບຈະມີໂນດທີ່ສະແດງສະຖານະຕ່າງໆ, ການຫາຄຳຕອບຈະອາໄສຫຼັກການຂອງເຄື່ອງຈຳກັດສະຖານະ (Finite State Machine) ໃນການດຳເນີນງານ

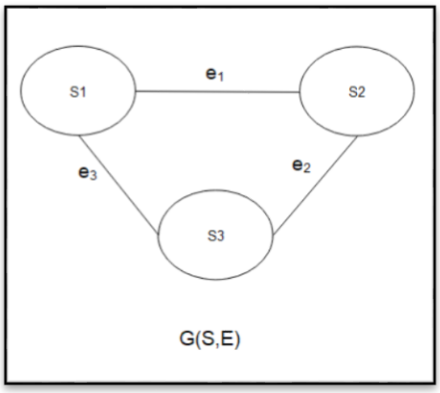
2.1.1 ກຣາບ (Graph)

ຄືໂຄງສ້າງຂໍ້ມູນແບບບໍ່ເປັນເສັ້ນຊື່ (Non-Linear) ທີ່ປະກອບມີ ໂນດ (Node) ຫຼື ສະຖານະ ແລະ ເສັ້ນເຊື່ອມ (Edge ຫຼື Link), ເຊິ່ງໂນດໝາຍເຖິງສິ່ງທີ່ສາມາດກຳນົດຊື່ ແລະ ເກັບຂໍ້ມູນໄດ້, ສຳລັບເສັ້ນເຊື່ອມແມ່ນເສັ້ນທີ່ເຊື່ອມກັນລະວ່າງໂນດ. ການກຳນົດທາງໄປຈາກໂນດໜຶ່ງໄປຫາອີກໂນດໜຶ່ງ ໂດຍທີ່ເສັ້ນທາງຈະບໍ່ຊ້ຳກັນ ໃນໂຄງສ້າງກຣາບຈະເອີ້ນວ່າ *ເສັ້ນທາງ* (Path). ກຣາບໜຶ່ງຈະມີໂນດພິເສດເອີ້ນວ່າ *ຮາກ* (Root), ເຊິ່ງຫມາຍເຖິງໂນດທີ່ມີເສັ້ນທາງຈາກໂຕມັນໄປຫາໂນດອື່ນຢູ່ເທິງກຣາບໄດ້ທຸກໂນດ, ຖ້າຫາກວ່າເສັ້ນທາງຈາກໂນດຮາກໄປຫາທຸກໂນດໃນກຣາບມີພຽງເສັ້ນທາງດຽວ ຈະເອີ້ນກຣາບນັ້ນ

ວ່າ ຕົ້ນໄມ້ (Tree) ແຕ່ຖ້າມີເສັ້ນທາງຈາກຮາກໄປຍັງໂນດອື່ນຫຼາຍກ່ວາໜຶ່ງເສັ້ນທາງຈະເອີ້ນໂຄງສ້າງນີ້ ວ່າ ກຣາບ.

ໂດຍທົ່ວໄປ ໃນໂຄງສ້າງຕົ້ນໄມ້ແລະກຣາບ ຈະຂຽນໃຫ້ໂນດຮາກຢູ່ເທິງສຸດ ຖ້າມີໂນດຢູ່ເທິງຂຶ້ນໄປ 1 ໂນດ ໂນດນັ້ນເອີ້ນວ່າ *ໂນດແມ່(* Parent Node*)* ໃນທາງດຽວກັນໂນດທີ່ຢູ່ລຸ່ມມັນເອີ້ນວ່າ *ໂນດລູກ (*Child Node*)* ແລະ ໂນດທັງໝົດທີ່ແຕກອອກມາຈາກໂນດແມ່ດຽວກັນເອີ້ນວ່າ *ໂນດພີ່ນ້ອງ* (Sibling Node*).* ຖ້າມີເສັ້ນທາງຈາກ X ໄປຫາໂນດ Y ເຮົາຈະເອີ້ນ X ວ່າເປັນ *ໂນດເທິງ* (Ancestor Node) ຂອງ Y ແລະ Y ເປັນ *ໂນດລຸ່ມ* (Descendant Node) ຂອງ X ສຳລັບໂນດອື່ນໆ ທີ່ເປັນຮາກຂອງຕົ້ນໄມ້ຍ່ອຍ (Subtree)ຈະປະກອບດ້ວຍຕົວຂອງມັນເອງ ແລະ ໂນດລຸ່ມທັງໝົດທີ່ຢູ່ກ້ອງໂຕມັນ.

ຖ້າກຳນົດສັນຍາລັກ G ແທນຄວາມໝາຍຂອງກຣາບ ອົງປະກອບທີ່ສຳຄັນຂອງ G ຈະປະກອບດ້ວຍກຸ່ມ ຂອງໂນດ ເຊິ່ງແທນດ້ວຍສັນຍາລັກ s ແລະກຸ່ມຂອງເສັ້ນເຊື່ອມ ເຊິ່ງແທນດ້ວຍສັນຍາລັກ E ແລະ ກຣາບ G ຈະແທນດ້ວຍ G (S, E) ຖ້າ $\{s_1, s_2, s_3, \ldots.\}$ຄືໂນດໃນກຸ່ມ s ແລະ $\{ e_1, e_2, e_3, \ldots.\}$ ຄື ເສັ້ນເຊື່ອມໃນກຸ່ມe ເຮົາສາມາດຂຽນຄວາມສຳພັນຂອງເສັ້ນເຊື່ອມແລະໂນດໄດ້ດັ່ງນີ້ $e_1= (s_1, s_2)$ເຊິ່ງໝາຍເຖິງ e_1 ຄືເສັ້ນເຊື່ອມຕໍ່ລະຫວ່າງໂນດ s_1ແລະ s_2 ແລະ ໂນດ s_1 ແລະ s_2 ຈະເອີ້ນວ່າ *ໂນດຂ້າງຄຽງ* (Adjacent Node) ດັ່ງຮູບທີ 2.2



ຮູບທີ 2.2 Node ແລະ ເສັ້ນເຊື່ອມຂອງ ກຣາບ G (S, E)

ຈຳນວນເສັ້ນທີ່ຕໍ່ມາຫາໂນດຈະເອີ້ນວ່າ *ດີກຣີ (Degree*) ຕົວຢ່າງ ໂນດ S_1 ຂອງກຣາບ G(S,E) ໃນຮູບທີ 2.2 ຈະມີເສັ້ນເຊື່ອມ e_1 ແລະ e_2 ຕໍ່ມາຮອດໂຕມັນ, ດັ່ງນັ້ນ ດີກຣີ ຂອງ s_1 ຈະເທົ່າກັບ 2 ເຊິ່ງເຮົາສາມາດຂຽນເປັນສັນຍາລັກໄດ້ແບບນີ້ $\deg(s_1) = 2$. ແຕ່ຖ້າໂນດໃດມີດີກຣີເປັນ 0 ເອີ້ນວ່າ *ໂນດດ່ຽວ (Isolated Node).*

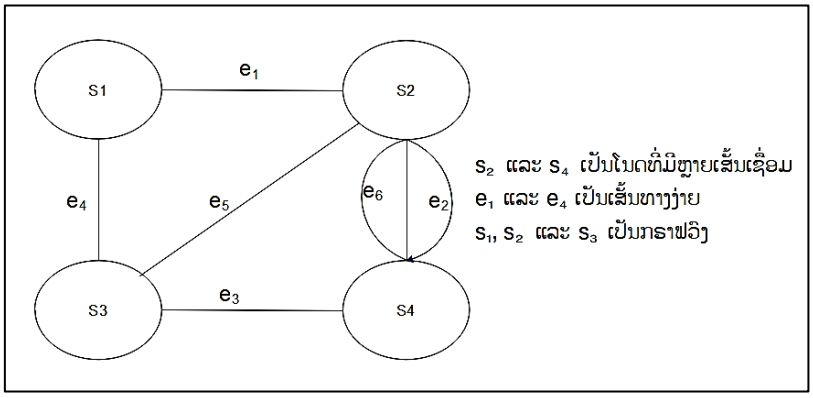
ຖ້າໃຫ້ p ໝາຍເຖິງເສັ້ນທາງຈາກໂນດ 1 ໄປຫາໂນດ 1 ຄວາມຍາວຂອງ p ຄືຈຳນວນເສັ້ນເຊື່ອມ e ທີ່ເຊື່ອມຢູ່ລະຫວ່າງທາງໄປຂອງໂນດ 2 ໂນດທີ່ກຳລັງພິຈາລະນາ ຫຼືເທົ່າກັບຈຳນວນໂນດລົບດ້ວຍ 1 ແລະເສັ້ນທາງສາມາດຂຽນແທນດ້ວຍສັນຍາລັກຕໍ່ໄປນີ້:

$$p = (s_0, s_1, s_2, s_3, \ldots, s_n)$$

ໝາຍເຖິງທາງໄປຈາກໂນດ s_0 ຫາໂນດ s_n ມີຄວາມຍາວເທົ່າກັບ n (ຈຳນວນໂນດທັງໝົດຄື n +1 ໂນດ) ແລະຈາກເສັ້ນທາງ p ຫາໂນດ s_0 ແລະ s_n ຄືໂນດດຽວກັນຈະເອີ້ນວ່າ *ເສັ້ນທາງປິດ (Close*

Path), ແຕ່ຖ້າໂນດທຸກໂນດຈາກ $s_0, \ldots, s_n$ ບໍ່ຊ້ຳກັນເສັ້ນທາງນີ້ເອີ້ນວ່າ *ເສັ້ນທາງງ່າຍ (Simple Path)* ແລະ ຖ້າເສັ້ນທາງງ່າຍແບບປິດມີຄວາມຍາວຕັ້ງແຕ່ 3 ຂຶ້ນໄປເອີ້ນວ່າ *ກຣາຟວົງ (Cycle Graph).*

ກຣາຟຈະຖືກເອີ້ນວ່າ *ກຣາຟເຊື່ອມ (Connected Graph)* ຖ້າທຸກໂນດໃນກຣາຟມີເສັ້ນທາງງ່າຍຕໍ່ເຖິງກັນໝົດ ແລະກຣາຟຈະຖືກເອີ້ນວ່າ *ກຣາຟສົມບູນ (Complete Graph)* ຖ້າທຸກໂນດໃນກຣາຟເປັນໂນດຂ້າງຄຽງກັນໝົດ ແລະໃນກຣາຟເຊື່ອມຕໍ່ຖ້າບໍ່ມີລູບເກີດຂຶ້ນຄືບໍ່ມີເສັ້ນທາງປິດກຣາຟນັ້ນຈະເປັນຕົ້ນໄມ້.



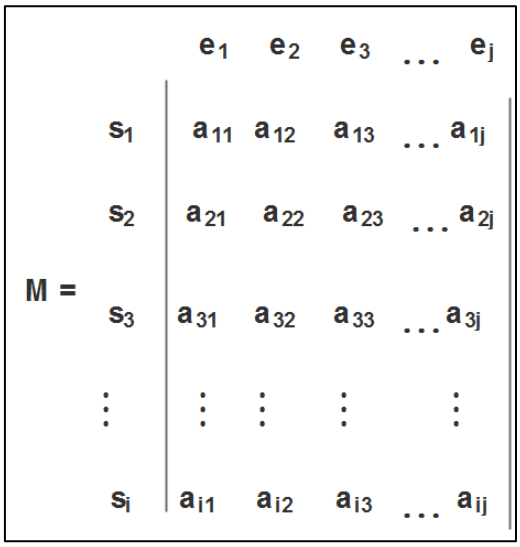
ຮູບ 2.3 ລັກສະນະຂອງເສັ້ນທາງໃນກຣາຟ

ຖ້າເສັ້ນເຊືອມໃນກຣາຟມີການລະບຸຂໍ້ມູນຈະເອີ້ນວ່າ *ກຣາຟປ້າຍ (Labelled Graph)* ແລະໃນກໍລະນີຂໍມູນທີ່ລະບຸເປັນຕົວເລກທີ່ບອກຂະໜາດຂອງຄວາມສຳຄັນຂອງເສັ້ນເຊື່ອມຈະເອີ້ນວ່າ *ນ້ຳໜັກ (Weight)* ໃນກໍລະນີທີ່ໂນດຄູ່ດຽວກັນ, ແຕ່ມີເສັ້ນເຊື່ອມຕໍ່ກັນຢູ່ຫຼາຍກ່ວາໜຶ່ງເສັ້ນເຊື່ອມຈະເອີ້ນເສັ້ນເຊື່ອມເຫຼົ່ານັ້ນວ່າ *ເສັ້ນເຊື່ອມຫຼາຍ (Multiple Edges)* ເຊິ່ງເສັ້ນເຊື່ອມດັ່ງກ່າວສາມາດຂຽນໄດ້ດັ່ງນີ້:

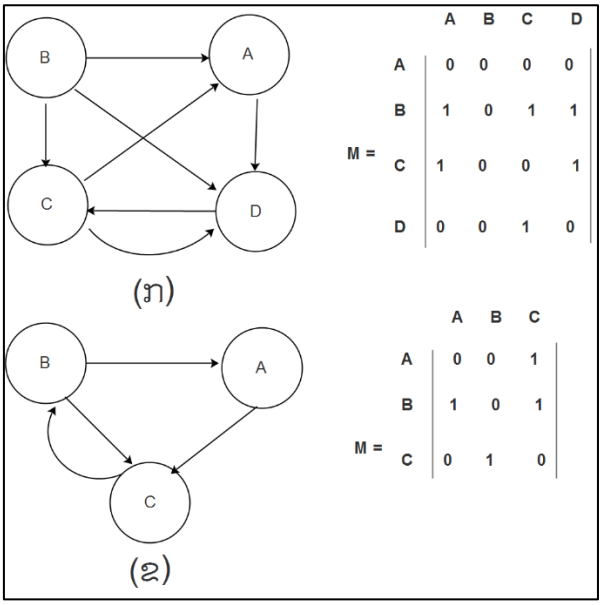
e_1 (s_1, s_2) ແລະ e_2 (s_1, s_2)

ຖ້າເສັ້ນເຊື່ອມທີ່ເກີດຂຶ້ນໃນກຣາຟມີການກຳນົດທິດທາງເຊັ່ນ ເລີ່ມຈາກໂນດ s_1 ໄປຫາ s_2 ກຣາຟນັ້ນຈະເອີ້ນວ່າ *ກຣາຟມີທິດ (Directed Graph)* ໃນການສະແດງຮູບແບບຂອງກຣາຟເພື່ອນຳມາໃຊ້ປະໂຫຍດນັ້ນມີຢູ່ຫຼາຍຮູບແບບ ແຕ່ຮູບແບບທີ່ນິຍົມກັນຫຼາຍຄືການໃຊ້ມາຕຣິດເອີ້ນວ່າ *ມາຕຣິດແບບໃກ້ກັນ (Adjacency Matrix)* ມາຕຣິດແບບໃກ້ກັນຄືມາຕຣິດທີ່ສ້າງຈາກກຣາຟ ໂດຍກຳນົດວ່າ ຖ້າ a_{ij} ເປັນອົງປະກອບ (Element) ໃນມາຕຣິດ Aຄ່າຂອງ a_{ij} ຈະເປັນດັ່ງນີ້

a_{ij} ຈະເທົ່າກັບ 1 ຖ້າ s_i ເປັນໂນດຂ້າງຄຽງກັບ s_j
a_{ij} ຈະເທົ່າກັບ 0 ຖ້າ s_i ແລະ s_j ບໍ່ເປັນໂນດຂ້າງຄ່ຽງກັນ



ຮູບທີ 2.4 ມາຕຣິດແບບໃກ້ກັນໃຊ້ແທນກຣາບ



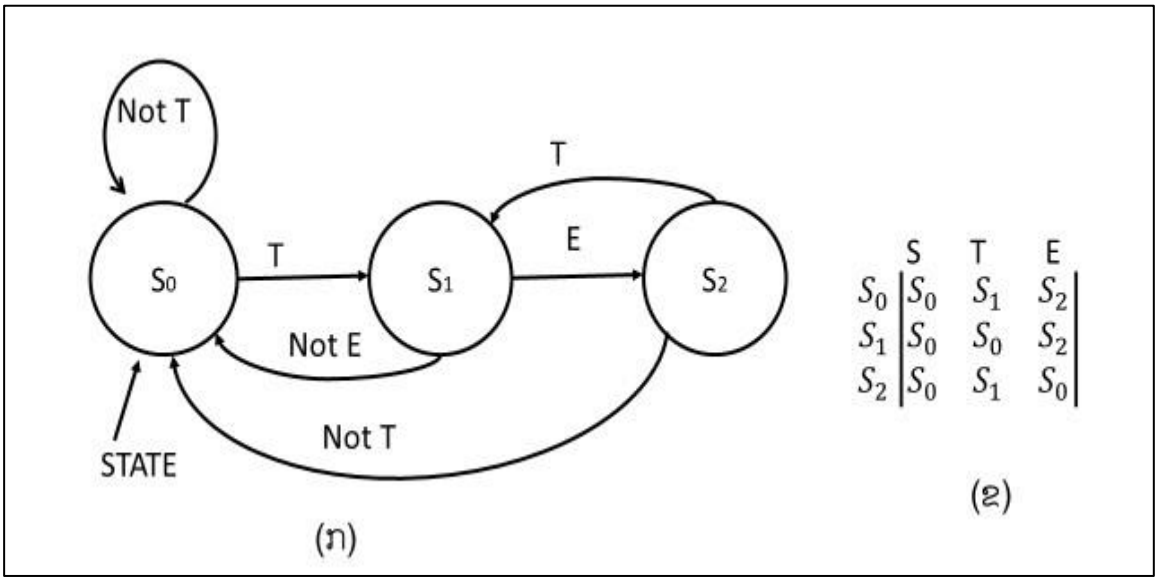
ຮູບທີ 2.5 ມາຕຣິດແບບໃກ້ກັນຂອງກຣາບ

2.1.2 ເຄື່ອງສະຖານະຈຳກັດ (Finite State Machine)

ເຄື່ອງສະຖານະຈຳກັດ ຫຼື ເຄື່ອງອັດຕະໂນມັດຈຳກັດ (Finite Automation)ເປັນວິທີການອະທິບາຍຮູບແບບຂອງພຶດຕິກຳ ເຊັ່ນ: ການປິດ-ເປີດປະຕູລິບ ແລະ ການເຮັດວຽກຂອງໂຕແບ່ງສ່ວນ (Parser) ເປັນຕົ້ນ, ທີ່ຂຽນແທນດ້ວຍກຣາບເຊື່ອມຕໍ່ຊະນິດກຣາບມີທິດ. ເຄື່ອງສະຖານະຈຳກັດມີອົງປະກອບ 3 ຢ່າງຄື: ຊຸດຂອງສະຖານະ ຫຼື ໂນດ, ຊຸດຂອງຄ່ານຳເຂົ້າ (Input Values) ແລະ ຟັງຊັນການປ່ຽນສະຖານະ (State Transition Function), ການເຮັດວຽກຂອງເຄື່ອງສະຖານະຈຳກັດຈະເລີ່ມຕົ້ນ (S_0) ເຊິ່ງເອີ້ນວ່າສະຖານະປັດຈຸບັນ, ເມື່ອມີຄ່ານຳເຂົ້າເຮັດວຽກກັບສະຖານະປັດຈຸບັນ ສະຖານະນີ້ກໍ່ຈະປ່ຽນໄປຕາມເງື່ອນໄຂຂອງຟັງຊັນການປ່ຽນສະຖານະ, ເຊິ່ງອາດຈະປ່ຽນສະຖານະໃໝ່ ຫຼື ບໍ່ປ່ຽນ

ສະຖານະ. ເມື່ອມີຄ່ານຳເຂົ້າຊຸດໃຫມ່ເຂົ້າມາເຮັດວຽກໃນສະຖານະປັດຈຸບັນນີ້ອີກ, ການປ່ຽນແປງກໍ່ຈະເກີດຕາມເງື່ອນໄຂຂອງຟັງຊັນການປ່ຽນສະຖານະອີກ ແລະ ເຮັດວົນໄປແບບນີ້ເລື່ອຍໆຈົນຈົບຂະບວນການ.

ເຄື່ອງສະຖານະຈຳກັດຊະນິດ *ຕົວຮັບ (Acceptor)* ເປັນເຄື່ອງສະຖານະຈຳກັດແບບງ່າຍຊະນິດໜຶ່ງ, ທີ່ຕອບຮັບ ຫຼື ປະຕິເສດຕໍ່ຄ່ານຳເຂົ້າເທົ່ານັ້ນ, ຖ້າຄ່ານຳເຂົ້າທັງໝົດທີ່ມີຕໍ່ສະຖານະປັດຈຸບັນໄດ້ຮັບການຕອບຮັບ, ຕ່ານຳເຂົ້າກໍ່ຈະໄດ້ຮັບການຍອມຮັບ ແລະ ຈະປ່ຽນສະຖານະປັດຈຸບັນໄປຢູ່ສະຖານະຖັດໄປ. ຖ້າສະຖານະປັດຈຸບັນຕອບປະຕິເສດ, ຄ່ານຳເຂົ້ານັ້ນກໍ່ຈະຖືກປະຕິເສດຄືກັນ. ຕົວຢ່າງຂອງເຄື່ອງສະຖານະຈຳກັດຊະນິດຕົວຮັບ ທີ່ມີການຄົ້ນຫາຕົວອັກສອນໃນຄຳ, ຖ້າເຮົາຈະຫາຕົວອັກສອນວ່າ “TE” ຈາກຄຳ “STATE” ລັກສະນະຂອງເຄື່ອງຈະເປັນດັ່ງສະແດງໃນຮູບ 2.6



ຮູບ 2.6 ເຄື່ອງສະຖານະຈຳກັດຊະນິດຕົວຮັບ

ຈາກຮູບ 2.6 ໂນດຕ່າງໆໃນກຣາບຈະສະແດງສະຖານະຂອງເຄື່ອງສະຖານະຈຳກັດ, ເຊິ່ງມີ s_0 ,s_1 ແລະ s_2 ເປັນສະຖານະ. ຄ່ານຳເຂົ້າຂອງລະບົບຄື “STATE” ເຊິ່ງປະກອບດ້ວຍຕົວອັກສອນ 5 ຕົວ, ຟັງຊັນການປ່ຽນສະຖານະຄືຕົວອັກສອນ “T” ສຳລັບສະຖານະ s_0 ແລະ ”E” ສຳລັບສະຖານະ s_1 ການເຮັດວຽກຂອງເຄື່ອງເລີ່ມຕົ້ນຈາກຄ່ານຳເຂົ້າ “STATE” ຕົວອັນສອນແຕ່ລະຕົວຂອງຄຳຈະຖືກສົ່ງເຂົ້າໄປໃນສະຖານະ s_0 ເທື່ອລະຕົວ, ຈາກຕົວທຳອິດຫາຕົວສຸດທ້າຍຂອງຄຳ. ຖ້າຕົວອັກສອນທີ່ຖືກສົ່ງເຂົ້າໄປບໍ່ແມ່ນ “T” ສະຖານະ s_0 ກໍ່ຈະຢູ່ຄົງທີ່, ແຕ່ຖ້າແມ່ນຈະຍ້າຍໄປສະຖານະໃໝ່ ຄື s_1 ແລະ ກວດສອບຕົວອັກສອນຕໍ່ໄປ. ຖ້າຄ່ານຳເຂົ້າຕໍ່ໄປເປັນ ”E” ການກວດສອບກໍ່ຈະສຳເລັດ, ແຕ່ຖ້າວ່າຄ່ານຳເຂົ້າຕົວໃຫມ່ບໍ່ແມ່ ”E” ສະຖານະໃຫມ່ຈະປ່ຽນໄປຢູ່ s_0 ແລະ ເລີ່ມຕົ້ນໃຫມ່, ຈາກນັ້ນຕົວອັກສອນໃຫມ່ຈະຖືກປ້ອນເຂົ້າມາຈົນຈົບຕົວອັກສອນສຸດທ້າຍຂອງຄຳ.

ຈາກຕົວຢ່າງຂ້າງເທິງ, ການເຮັດວຽກແບບລະອຽດຕະເລີ່ມຈາກຕົວອັກສອນທຳອິດ “S” ຖືກສົ່ງເຂົ້າມາທີ່ s_0 ເນື່ອງຈາກ s_0 ບໍ່ຄືກັບຟັງຊັນການປ່ຽນສະຖານະ, ດັ່ງນັ້ນຕົວອັກສອນ “S” ຈະຖືກປະຕິເສດ ແລະ ສະຖານະປັດຈຸບັນຍັງຢູ່ທີ່ s_0 , ຈາກນັ້ນຕົວອັກສອນໃໝ່ຈະຖືກສົ່ງເຂົ້າມາທີ່ສະຖານະ s_0 ຄື “T”

ແລະໄດ້ຮັບການຕອບຮັບເຮັດໃຫ້ສະຖານະປັດຈຸປັນຍ້າຍໄປເປັນສະຖານະໃໝ່ s_1 ແລະ ຕົວອັນສອນໃໝ່ ຈະຖືກສົ່ງເຂົ້າມາໃນສະຖານະ s_1 ຄື "A" ເນື່ອງຈາກຟັງຊັນການປ່ຽນສະຖານະຄື "E" ເຮັດໃຫ້ສະຖານະໃໝ່ຂອງສະຖານະປັດຈຸປັນຍ້າຍໄປທີ່ s_0 ຄືນອີກ ແລະ ຕົວອັນສອນໃໝ່ "T" ກໍ່ຈະຜ່ານຈາກ s_0 ໄປ s_1 , ຈາກນັ້ນ "E" ກໍ່ຈະເຂົ້າມາເຮັດໃຫ້ສະຖານະຍ້າຍຈາກ s_1 ໄປ s_2 ແລະສິ້ນສຸດການເຮັດວຽກ.

ການປ່ຽນສະຖານະເຄື່ອງສະຖານະຈຳກັດສາມາດຂຽນໃນຮູບແບບຂອງມາຕຣິດໄດ້ດັ່ງສະແດງໃນຮູບ 2.6 (ຂ) ເຊິ່ງເອີ້ນວ່າ *ທຣານຊິເຊັນມາຕຣິດ (Transition Matrix).*

ນິຍາມທາງຄະນິດສາດຂອງເຄື່ອງສະຖານະຈຳກັດສາມາດຂຽນໄດ້ດັ່ງນີ້:

ເຄື່ອງສະຖານະຈຳກັດຊະນິດຕົວຮັບ ສັນຍາລັກ 4 ຕົວຄື: (Σ, S, S_0, Δ) ເຊິ່ງ

- Σ ຄືຄ່ານຳເຂົ້າ
- S ເປັນກຸ່ມຂອງສະຖານະ
- S_0 ເປັນສະຖານະຕ່າງໆໃນກຸ່ມຂອງ S
- Δ ເປັນຟັງຊັນການປ່ຽນສະຖານະເທົ່າກັບ $\Delta : S * \Sigma \rightarrow S$

2.1.3 ການກຳນົດນິຍາມໃຫ້ກັບປັນຫາ

ເພື່ອເປັນການທຳຄວາມເຂົ້າໃຈໃນເຕັກນິກຂອງປັນຍາປະດິດເບື້ອງຕົ້ນ ໃນທີ່ນີ້ຈະສະແດງໃຫ້ເຫັນເຖິງວິທີການແກ້ປັນຫາ ແລະຫຼັກການບາງຢ່າງໃນການກຳນົດຫຼັກເກນຂອງປັນຫາ ວິທີການແລະຫຼັກການດັ່ງກ່າວເປັນຂະບວນການພື້ນຖານທີ່ຈະທຳໃຫ້ເຂົ້າໃຈການແກ້ປັນຫາ (problem solving) ທາງດ້ານຄອມພິວເຕີ ກຳນົດໃຫ້ມີໂຖ 2 ໃບທີ່ມີຂະໜາດໃສ່ນ້ຳ 4 ແລະ 3 ລິດຕາມລຳດັບ ຈະມີວິທີການແນວໃດເພື່ອເຮັດໃຫ້ໂຖນ້ຳທີ່ມີຂະໜາດ 4 ລິດ ມີນ້ຳຢູ່ 2 ລິດພໍດີ ໂດຍທີ່ບໍ່ອາໄສເຄື່ອງວັດແທກໃດໆຈາກໂຈດດັ່ງກ່າວຂ້າງຕົ້ນ ການແກ້ປັນຫາມີຫຼາຍວິທີ ແຕ່ໃນທີ່ນີ້ຈະອາໄສວິທີການຂອງ ຂອບເຂ ສະຖານະ (state space) ມາຊ່ວຍໃນການແກ້ ໂດຍທີ່ກຳນົດຈຸດເລີ່ມຕົ້ນຂອງປັນຫາ ແລະ ເປົ້າໝາຍຂອງການແກ້ປັນຫາວ່າຄືຫຍັງ ສຳລັບເລື່ອງໂຖນ້ຳ ການແກ້ປັນຫາຈະເລີ່ມຕົ້ນຈາກ ໂຖນ້ຳ 2 ໂຕທີ່ບໍ່ມີນ້ຳເລີຍ ແລະ ຜົນຮັບສຸດທ້າຍ ຈະຕ້ອງເປັນ ມີນ້ຳ 2 ລິດໃນໂຖທຳອິດ ແລະ ໂຖທີ່ 2 ມີນ້ຳເທົ່າໃດກໍ່ໄດ້. ຖ້າກຳນົດວ່າ X ເຊິ່ງເປັນຕົວປ່ຽນໃດ ໆ ທີ່ແທນຄ່າຂອງປະລິມານນ້ຳໃນໂຖນ້ຳທຳອິດ ແລະ Y ຄືຕົວປ່ຽນໃດ ໆ ທີ່ແທນຄ່າຂອງປະລິມານນ້ຳໃນໂຖນ້ຳທີ່ 2 ດັ່ງນັ້ນຄ່າຕ່າງໆ ທີ່ເປັນໄປໄດ້ທັງໝົດຂອງ X ແລະ Y ຈະເປັນດັ່ງນີ້

X = 0, 1, 2, 3, 4 (ເນື່ອງຈາກໂຖນ້ຳທຳອິດປັນຈຸນ້ຳໄດ້ 4 ລິດ) ແລະ

Y = 0, 1, 2, ຫຼື 3 (ເນື່ອງຈາກໂຖນ້ຳທີ່ສອງປັນຈຸນ້ຳໄດ້ 3 ລິດ)

ແລະປະລິມານນ້ຳໃນໂຖນ້ຳ 2 ໂຖ ສາມາດຂຽນຢູ່ໃນຮູບຂອງ X ແລະ Y ໄດ້ເປັນ (X,Y) ຈຸດເລີ່ມຕົ້ນໃນການແກ້ປັນຫາເລີ່ມຈາກ (0,0) ເຊິ່ງໝາຍເຖິງສະຖານະທີ່ບໍ່ມີນ້ຳໃນໂຖນ້ຳ ສຳລັບເປົ້າໝາຍທີ່ຕ້ອງການຄື (2,n) ໂດຍ n ເປັນເລກໃດໆ ທີ່ມີຄ່ານ້ອຍກວ່າ 3ໃນການແກ້ປັນຫາ ຈະຕ້ອງທຳການປ່ຽນສະຖານະ ຈາກ (0,0) ໄປເປັນ (2,n) ເຊິ່ງຈະເຮັດໄດ້ໂດຍການອາໄສກົດຕ່າງໆທີ່ກ່ຽວກັບການເທນ້ຳຂອງປັນຫານີ້, ກົດເຫຼົ່ານີ້ໄດ້ແກ່

1. ຕື່ມນ້ຳລົງໃນໂຖທຳອິດຈົນເຕັມ

2. ເຕີມນ້ຳລົງໃນໂຖໃບສອງຈົນເຕັມ
3. ເທນ້ຳບາງສ່ວນອອກຈາກໂຖທຳອິດ
4. ເທນ້ຳບາງສ່ວນອອກຈາກໂຖສອງ

ຖ້າຫາກອະທິບາຍກົດເຫຼົ່ານີ້ໃນຮູບຂອງຕົວປ່ຽນ X ແລະ Y ຈະໄດ້ດັ່ງຕາຕະລາງທີ 2.1

ລຳດັບ	ກົດເກນ	ຄຳອະທິບາຍ
1	(X,Y: X < 4) -> (4,Y)	ຖອກນ້ຳໃສ່ໂຖທຳອິດຈົນເຕັມ
2	(X,Y: Y < 3) -> (X,3)	ຖອກນ້ຳໃສ່ໂຖທີ 2ຈົນເຕັມ
3	(X,Y: X > D) -> (X-D,Y)	ຖອກນ້ຳບາງສ່ວນອອກຈາກໂຖ 2
4	(X,Y: Y > D) -> (X,Y-D)	ຖອກນ້ຳອອກຈາກໂຖທຳອິດຈົນໝົດ
5	(X,Y: Y > 0) -> (X,0)	ຖອກນ້ຳອອກຈາກໂຖທີ 2 ຈົນໝົດ
6	(X,Y: X+Y >= 4^Y > 0) -> (4, Y-(4-X))	ຖອກນ້ຳອອກຈາກໂຖທີ 2 ໃສ່ໂຖທຳອິດ ຈົນເຕັມ ໂດຍໂຖທີ 1 ມີນ້ຳລວມກັນ ຫຼາຍກວ່າ 4 ລິດ
7	(X,Y: X+Y >= 3^X > 0) -> (X-(3-Y),3)	ຖອກນ້ຳອອກຈາກໂຖທຳອິດ ໃສ່ໂຖທີ 2 ຈົນເຕັມ ໂດຍ2 ໂຖ ມີນ້ຳລວມກັນຫຼາຍ ກວ່າ 3 ລິດ
8	(X,Y: X+Y <= 4^Y > 0) -> (X+Y,0)	ຖອກນ້ຳອອກຈາກໂຖທີ2 ໃສ່ໂຖທຳອິດ ຈົນເຕັມໂດຍ 2 ໂຖມີນ້ຳລວມກັນຫຼາຍ ກວ່າ 4 ລິດ
9	(X,Y: X+Y <= 3^X > 0) -> (0,X+Y)	ຖອກນ້ຳອອກຈາກໂຖທີ 2ໃສ່ໂຖທີ 1
10	(X>0,X+Y≤3) → (0,X+Y)	ຖອກນ້ຳອອກຈາກໂຖທຳອິດ ໃສ່ໂຖທີ 2

ຕາຕະລາງທີ 2.1 ກົດທີ່ອະທິບາຍການແກ້ບັນຫາເລື່ອງໂຖນ້ຳ

ໃນການກຳນົດນິຍາມຂອງປັນຫານັ້ນມີສິ່ງທີ່ຄວນເອົາໃຈໃສດັ່ງນີ້

1. ໃນການຂຽນກົດເພື່ອອະທິບາຍປັນຫາໜຶ່ງຈະຕ້ອງມີອົງປະກອບພື້ນຖານຢ່າງໜ້ອຍ 2 ສ່ວນທີ່ໃຊ້ກວດ ສອບເງື່ອນໄຂ (condition) ແລະ ຂໍ້ສະຫຼຸບ (conclusion). ສ່ວນທີ່ເປັນເງື່ອນໄຂຄືສ່ວນທີ່ໃຊ້ກວດສອບ ເພື່ອນຳກົດຂໍ້ນັ້ນມາໃຊ້ ແລະ ສ່ວນທີ່ເປັນຂໍ້ສະຫຼຸບແມ່ນສ່ວນທີ່ນຳໃຊ້ກົດ ເຊັ່ນ:

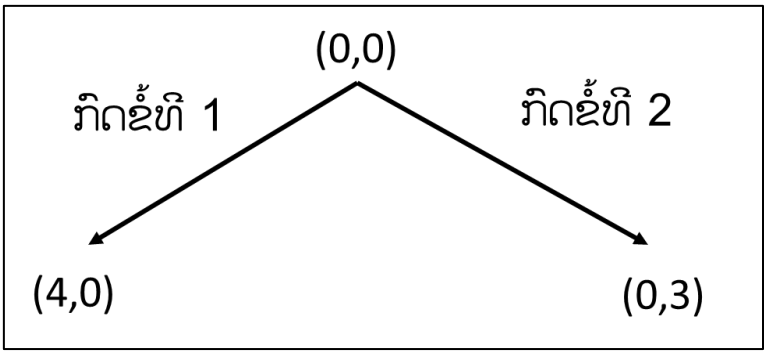
(X<4,Y) -> (4,Y)

ໝາຍຄວາມວ່າ ຖ້າ X ນ້ອຍກວ່າ 4 ລິດ (X<4,Y) ຈະສະຫຼຸບວ່າເຮົາສາມາດເຮັດໃຫ້ມີນ້ຳຢູ່ 4 ລິດໃນ ໂຖນ້ຳທຳອິດໄດ້ ແລະມີນ້ຳຈຳນວນ Y ໃນໂຖນ້ຳທີ່ສອງຜົນທີ່ໄດ້ອອກມາຈະເປັນ (4,y)
ລັກສະນະຂອງກົດທີ່ສະແດງຂ້າງເທິງ ເອີ້ນວ່າ *ກົດການຜະລິດ (production rule)* ເຊິ່ງກົດການຜະລິດນີ້ ຈະຕ້ອງ
ກຳນົດຫຼືສ້າງກົດຈຳນວນໜຶ່ງທີ່ສາມາດອະທິບາຍການເຮັດວຽກທັງໝົດທີ່ຈຳເປັນຕໍ່ການແກ້ປັນຫານັ້ນ ໆ
2. ການກຳນົດຄ່າຂອງ X ແລະ Y ຈະຕ້ອງສາມາດຄອບຄລຸມເຖິງຄວາມເປັນໄປໄດ້ທັງໝົດ ເຊິ່ງຄ່າຂອງ X
ແລະ Y ນີ້ຈະເປັນຕົວໃນການກຳນົດສະຖານະ (state) ຕ່າງໆ ຂອງການແກ້ປັນຫາ ໃນທີ່ນີ້ເຮົາກຳນົດ ສະຖານະຕ່າງໆ ເປັນ
(X,Y) ໂດຍມີຄ່າ X = 0, 1, 2, 3, 4 ແລະ y = 0, 1, 2, 3
3. ໃນການກຳນົດຂັ້ນຕອນຕ່າງໆ ຂອງການແກ້ປັນຫາເອີ້ນວ່າ ຂອບເຂດສະຖານະ (state space), ເຊິ່ງ ກຳນົດວ່າຂອບເຂດ(space) ໜຶ່ງໆ ຢ່າງນ້ອຍຈະຕ້ອງມີ 2 ສະຖານະຄື ສະຖານະເລີ່ມຕົ້ນ (Start State) ເປັນ (0,0) ແລະ ສະຖານະເປົ້າໝາຍ(Goal State) ເປັນ (2,n) ການແກ້ປັນຫາເປັນການນຳກົດ ມາ ອະທິບາຍການປ່ຽນແປງຕ່າງໆ ຂອງສະຖານະ ຈາກສະຖານະເລີ່ມຕົ້ນຈົນເຖິງສະຖານະເປົ້າໝາຍ ໃນ ການແກ້ປັນຫາຕາມລັກສະນະນີ້ຈະເອີ້ນວ່າ ລະບົບການຜະລິດ (production system). ຈາກຕົວຢ່າງຂ້າງ ຕົ້ນ ສະຖານະຂອງປັນຫາເລີ່ມຕົ້ນຄື (0,0) ເຮົາຈະຕ້ອງຫາທາງປ່ຽນຈາກ (0,0) ໃຫ້ເປັນ
(2,n) ການປ່ຽນເລີ່ມຈາກການສຳຫຼວດກົດ ຈາກຂໍ້ 1 ເຖິງ 8 ຂ້າງເທິງ ວ່າມີກົດຂໍ້ໃດສາມາດໃຊ້ກັບ ສະຖານະເລີ່ມຕົ້ນໄດ້

1. (X<4,Y) → (4,Y)
2. (X,Y<3) →(X,3)
3. (X>0,Y) →(0,Y)
4. (X,Y>0) → (X,0)
5. (X+Y≥4,Y>0) → (4,Y-(4-X))
6. (X>0,X+Y≥3) → (X-(3-Y),3)
7. (X+Y≤4,Y>0) → (X=Y,0)
8. (X>0,X+Y≤3) → (0,X+Y)

ຈາກການສຳຫຼວດມີກົດຂໍ້ 1 ແລະ 2 ເທົ່ານັ້ນທີ່ສາມາດນຳມາໃຊ້ກັບເງື່ອນໄຂນີ້ໄດ້ ເພາະວ່າໃນ ສ່ວນຂອງເງື່ອນໄຂ
ຄື (X<4,Y) ແລະ (X,Y<3) ຕາມລຳດັບ ເປັນຈິງຕາມສະຖານະເລີ່ມຕົ້ນ (0,0) ຄື ຄ່າຂອງ X<4 ຕາມເງື່ອນໄຂ
ຂອງກົດຂໍ້ 1 ແລະ Y<3 ຕາມເງື່ອນໄຂຂອງກົດຂໍ້ 2 ດັ່ງນັ້ນຈາກສະຖານະ (0,0) ເຮົາສາມາດປ່ຽນ ເປັນ(4,0)
ດ້ວຍກົດຂໍ້ 1 ເພາະກົດຂໍ້ 1 ບອກວ່າຖ້າ X<4 ສາມາດເຮັດໃຫ້ X ເປັນ 4 ແລະ Y ຄົງທີ່ໄດ້ ແລະເຮົາ

ສາມາດປ່ຽນຈາກ
(0,0) ເປັນ (0,3) ໄດ້ໂດຍກົດຂໍ້ທີ 2 ເຊິ່ງສາມາດຂຽນເປັນຮູບໄດ້ດັ່ງນີ້



ຮູບທີ 2.7 ການປ່ຽນສະຖານະຈາກ (0,0) ເປັນ (4,0) ແລະ (0,3)

ຈາກນັ້ນສ້າງການສຳຫຼວດສະຖານະ (4,0) ກໍພົບວ່າມີກົດຂໍ້ 2, 3 ແລະ 6 ທີ່ມີເງື່ອນໄຂກົງກັນ ແລະ ສະຖານະຕ່າງໆ ຈະປ່ຽນໄປດັ່ງນີ້

(4,0) -> (4,3) ໂດຍການໃຊ້ກົດຂໍ້ 2

(4,0) -> (0,0) ໂດຍການໃຊ້ກົດຂໍ້ 3

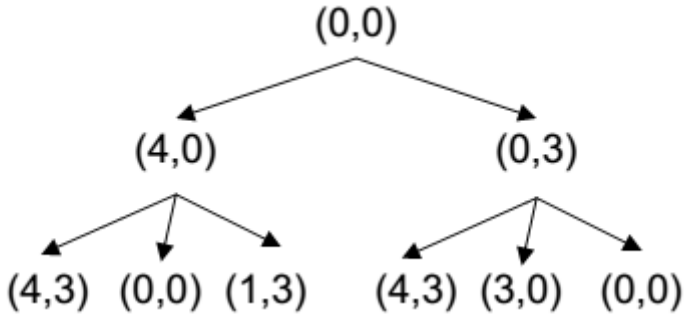
(4,0) -> (1,3) ໂດຍການໃຊ້ກົດຂໍ້ 6

ຈາກການສຳຫຼວດສະຖານະ (0,3) ມີກົດຂໍ້ 1, 4 ແລະ 7 ທີ່ມີເງື່ອນໄຂກົງກັນ ແລະສະຖານະຕ່າງໆ ທີ່
ເກີດໃໝ່ຈະໄດ້ເປັນດັ່ງນີ້

(0,3) -> (4,3) ໂດຍກົດຂໍ້ 1

(0,3) -> (0,0) ໂດຍກົດຂໍ້ 4

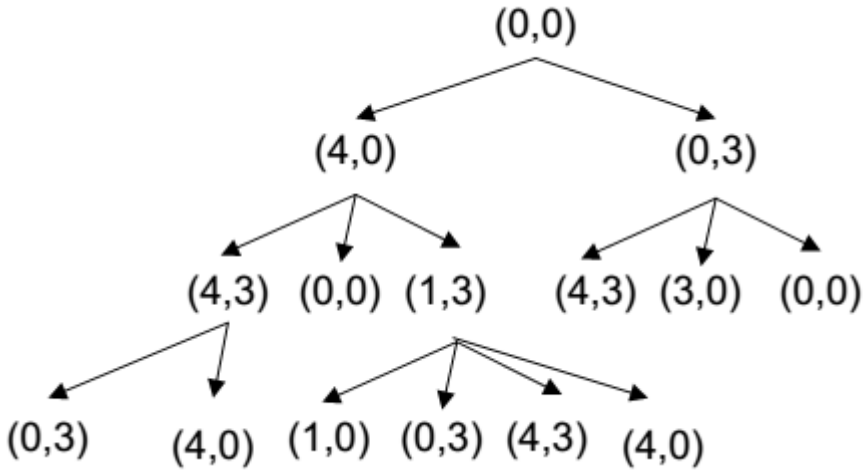
(0,3) -> (3,0)



ຮູບທີ 2.8 ການຂະຫຍາຍໂນດມາໃນລະດັບທີ່ສອງ

ຈາກນັ້ນໄປຂະຫຍາຍໂນດ (4,3) ທີ່ຢູ່ແຖວລຸ່ມສຸດແລະຢູ່ຊ້າຍມືສຸດຕາມຮູບທີ່ 2.8 ແລ້ວຫາກົດທີ່ທີ່ເໝາະສົມມາໃຊ້ແລ້ວຂ້າມໄປພິຈາລະນາ (1,3) ແລະ (3,0) ຕາມລຳດັບ ເຮົາຈະເຫັນໄດ້ວ່າໂນດ

(0,0) (4,3) ແລະ (0,0)ຈະບໍ່ພິຈາລະນາເພາະເປັນໂນດທີ່ຊ້ຳກັບໂນດທີ່ມີຢູ່ແລ້ວ ເມື່ອພິຈາລະນາຈົນຄົບ ແລະ ບໍ່ ພິຈາລະນາໂນດຊ້ຳ ເຮົາຈະໄດ້ໂນດທັງໝົດດັ່ງນີ້



ຮູບທີ 2.9 ຂອບເຂດປັນຫາຂອງການແກ້ປັນຫາ

ຈາກຕົ້ນໄມ້ຂອງຮູບຂ້າງເທິງ ຈະເຫັນວ່າມີຢູ່ຫຼາຍເສັ້ນທາງທີ່ສາມາດແກ້ປັນຫານີ້ໄດ້ ຖ້າເຮົາຈະເລືອກເສັ້ນທາງໜຶ່ງຂອງການແກ້ປັນຫາສາມາດຂຽນເປັນຕາຕະລາງຂອງການແກ້ປັນຫາໄດ້ດັ່ງນີ້

ໂຖ 1	ໂຖ 2	ກົດຂໍ້ທີ	ສະຖານະ
0	0		ເລີ່ມຕົ້ນ
0	3	2	
3	0	7	
3	3	2	
4	2	5	
0	2	3	
2	0	7	ເປົ້າໝາຍ

ຮູບທີ 2.9 ສະແດງການແກ້ປັນຫາເລື່ອງໂຖນ້ຳ

ການແກ້ປັນຫາແບບທີ່ກ່າວມາແລ້ວເປັນຕົວຢ່າງໆຂອງການແກ້ປັນຫາແບບ ລະບົບການຜະລິດ (Production) ເຊິ່ງເປັນຂະບວນການໃນການວາງໂຄງສ້າງໂປຣແກຣມແບບປັນຍາປະດິດ ໃຫ້ມີລັກສະນະງ່າຍຕໍ່ການອະທິບາຍ ຂະບວນການ (ຫຼື ວິທີການແກ້ປັນຫາ) ການແກ້ປັນຫາແບບນີ້ຈະຕ້ອງປະກອບໄປດ້ວຍ

1. ກົດເຊິ່ງກົດແຕ່ລະຂໍ້ຈະຕ້ອງປະກອບດ້ວຍສ່ວນທີ່ຢູ່ທາງຊ້າຍທີ່ຈະອະທິບາຍເຖິງເງື່ອນໄຂຂອງກົດ ແລະ ສ່ວນທີ່ຢູ່ທາງຂວາທີ່ອະທິບາຍເຖິງຜົນຂອງກົດ

2. ຖານຂໍ້ມູນທີ່ມີການສະຫຼຸບທີ່ຕ້ອງການ ບາງສ່ວນຂອງຖານຂໍ້ມູນຈະເປັນແບບຖາວອນ ແລະບາງສ່ວນຈະເປັນສິ່ງທີ່ກ່ຽວຂ້ອງກັບການແກ້ປັນຫາໃນຊ່ວງນັ້ນ

3. ກົນໄກໃນການຄວບຄຸມ ເປັນສ່ວນທີ່ກຳນົດລຳດັບຂອງກົດທີ່ຈະນຳມາໃຊ້ຫຼືປຽບທຽບກັບຖານຂໍ້ມູນ ເພື່ອບອກລຳດັບຂອງການແກ້ປັນຫາ ສຳລັບກົນໄກໃນການຄວບຄຸມນີ້ມີຢູ່ 3 ສິ່ງທີ່ສຳຄັນຄື:

- ການກຳນົດທິດທາງສຳລັບການຄົ້ນຫາ
- ຂະບວນການໃນການເລືອກກົດເກນ ແລະ
- ການຄົ້ນຫາແບບຮິວຣິສະຕິກ

2.1.4 ວິທີການຕ່າງໆ ໃນການແກ້ປັນຫາ

ຈາກຕົວຢ່າງທີ່ກ່າວເຖິງຂະບວນການໃນການເລືອກແລະຈັດລຽງລຳດັບກົດນັ້ນ ເປັນຂະບວນການໜຶ່ງຂອງການແກ້ປັນຫາ ຫຼືຖ້າຈະເບິ່ງອີກແບບໜຶ່ງກໍຄືການຫາເຫດຜົນ ໃນການແກ້ປັນຫາຫຼືການຫາເຫດຜົນນີ້. ຂະບວນການຂອງຄອມພິວເຕີທີ່ຈະຕ້ອງສ້າງຄືການຄົ້ນຫາຂໍ້ມູນ (Search) ດັ່ງຕົວຢ່າງເລື່ອງ ໂຖນ້ຳ 2 ໂຖນັ້ນ ເປັນພຽງຕົວຢ່າງຂອງການສະແດງໃຫ້ເຫັນເຖິງວິທີການຂອງການຄົ້ນຫາຂໍ້ມູນທີ່ຍັງມີຄວາມຮູ້ບໍ່ຫຼາຍເທົ່າໃດ ຖ້າຫາກເບິ່ງປັນຫາທີ່ໃຫຍ່ຂຶ້ນ ເຊັ່ນ ການຫຼິ້ນໝາກລຸກ, ໃນການຫຼິ້ນໝາກລຸກນັ້ນ ການໄປໝາກລຸກໃນເທື່ອໜຶ່ງໆ ຜູ້ຫຼິ້ນຈະຕ້ອງພິຈາລະນາຈາກຄວາມເປັນໄປໄດ້ທັງໝົດວ່ະເລ່ຍແລ້ວ ປະມານ 35 ຕາ ເພື່ອທີ່ຈະເລືອກໄປພຽງຕາດຽວ ແລະໃນເກມໜຶ່ງໆ ຜູ້ຫຼິ້ນຈະຕ້ອງໄປໂດຍສະເລ່ຍ 50 ເທື່ອເຖິງຈະຮູ້ຜົນແພ້ຊະນະ ດັ່ງນັ້ນໃນການໄປຂອງໝາກລຸກທັງກະດານຜູ້ຫຼິ້ນຈະຕ້ອງພິຈາລະນາເຖິງ 35^{2x50}

ເຊິ່ງເປັນເລື່ອງຍາກຫຼາຍທີ່ຈະອາໄສການຄົ້ນຫາຄຳຕອບດ້ວຍວິທີການຂອງການຄົ້ນຫາຂໍ້ມູນທຳມະດາ ໃນການແກ້ປັນຫາຫາຂອງປັນຍາປະດິດນັ້ນ ມີເລື່ອງໃຫຍ່ທີ່ຈະຕ້ອງພິຈາລະນາເຖິງຄື:

- ການກຳໜົດທິດທາງສຳລັບການຄົ້ນຫາ ແລະ ຮູບແບບຂອງໂຄງສ້າງຂໍ້ມູນທີ່ໃຊ້ສຳລັບການຄົ້ນຫາ
- ການສະແດງຄວາມຮູ້
- ຂະບວນການໃນການເລືອກກົດເກນ ແລະ
- ການຄົ້ນຫາແບບຮິວຣິສະຕິກການກຳນົດທິດທາງສຳລັບການຄົ້ນຫາ ແລະຮູບແບບຂອງໂຄງສ້າງຂໍ້ມູນທີ່ໃຊ້ສຳລັບການຄົ້ນຫາ ເປັນການກຳນົດລຳດັບຂອງການພິຈາລະນາໂນດຫຼືສະຖານະຕ່າງໆ ໃນໂຄງສ້າງຂໍ້ມູນສຳລັບການຄົ້ນຫາຄຳຕອບ ຈາກການແກ້ໄຂປັນຫາເລື່ອງໂຖນ້ຳເຮົາຈະເຫັນວ່າລຳດັບຂອງການແກ້ປັນຫາຈະເລີ່ມຕົ້ນຈາກ (0,0) ເຊິ່ງເປັນສະຖານະເລີ່ມຕົ້ນຈາກນັ້ນເຮົາກໍສ້າງໂນດລູກອອກມາ 2 ໂນດຄື (0,4) ແລະ (0,3) ຈາກນັ້ນກໍຈະພິຈາລະນາໂນດ (0,4) ແລະ (0,3) ຕາມລຳດັບ ແລະສ້າງໂນດລູກອອກມາ ເປັນ (4,3) (0,0) (1,3) (4,3) (3,0) ແລະ (0,0) ແລະເຮົາກໍຈະພິຈາລະນາໂນດລູກເຫຼົ່ານີ້ ຕາມລຳດັບ ການພິຈາລະນາລຳດັບໃນລັກສະນະນີ້ຈະເປັນແບບ ການພິຈາລະນາແບບລວງກວ້າງກ່ອນ (Breath First Search) ໃນການກຳນົດລຳດັບຂອງໂນດທີ່ເຮົາຈະພິຈາລະນານອກຈາກການພິຈາລະນາແບບລວງກວ້າງກ່ອນ ຍັງມີວິທີອື່ນ ໆ ອີກເຊັ່ນ *ການຄົ້ນຫາແບບເລິກກ່ອນ (Depth First Search)* , *ການຄົ້ນຫາຈາກໂນດທີ່ດີທີ່ສຸດກ່ອນ (Best First Search)* ເປັນຕົ້ນ, ເຊິ່ງການຄົ້ນຫາແບບຕ່າງໆ ເຫຼົ່ານີ້

ຈະໄດ້ກ່າວຢ່າງລະອຽດໃນຂໍ້ຕໍ່ໄປ. ສຳລັບໂຄງສ້າງຂໍ້ມູນ ໂດຍສ່ວນໃຫຍ່ແລ້ວຈະຖືກກຳນົດໃຫ້ຢູ່ໃນຮູບ ຂອງໂຄງສ້າງຕົ້ນໄມ້ ຫຼື ໂຄງສ້າງກຣາບ ເຊິ່ງໂຄງສ້າງທັງສອງນີ້ສາມາດປ່ຽນແປງໄປມາໄດ້ ຕົວຢ່າງ ຂອງໂຄງສ້າງຂໍ້ມູນແບບຕົ້ນໄມ້ ແລະກຣາບໄດ້ສະແດງໄວ້ດັ່ງ ຮູບທີ່ 3 ເຊິ່ງເປັນເລື່ອງຂອງໂຖນ້ຳ. ການ ສ້າງຮູບແບບເຄືອຄ່າຍຂໍ້ມູນແບບກຣາບໂດຍທົ່ວໄປແລ້ວຈະດີກວ່າແບບຕົ້ນໄມ້ ເນື່ອງຈາກຈະເຮັດໃຫ້ປະ ຢັດເນື້ອທີ່ຂອງໜ່ວຍຄວາມຈຳແລະການຄົ້ນຫາຂໍ້ມູນເຮັດໄດ້ໄວກວ່າ ແຕ່ຂໍ້ເສຍກໍຄືທຳການເຊື່ອມຕໍ່ຂອງຂໍ້ ມູນເຮັດໄດ້ຍາກແລະໃຊ້ເວລາໃນການເຊື່ອມຕໍ່ຂໍ້ມູນດົນກວ່າ ດັ່ງນັ້ນໃນການອອກແບບໂຄງສ້າງບາງເທື່ອ ຈຶ່ງນິຍົມສ້າງເປັນແບບຕົ້ນໄມ້ກ່ອນແລ້ວຍຈຶ່ງປ່ຽນເປັນກຣາບການປ່ຽນຂະບວນການຄົ້ນຫາໃນໂຄງສ້າງ ຕົ້ນໄມ້ (Tree Search Procedure) ເປັນຂະບວນການຄົ້ນຫາໃນໂຄງສ້າງກຣາບ (Graph Search Procedure)ສາມາດເຮັດໄດ້ໂດຍວິທີການດັ່ງຕໍ່ໄປນີ້:

1.ກວດສອບກຸ່ມຂອງໂນດທີ່ໄດ້ສ້າງມາແລ້ວ່າໂນດທີ່ຈະສ້າງໃໝ່ມາແລ້ວຫຼືບໍ່

2. ຖ້າຍັງບໍ່ມີ ກໍ່ຈະສ້າງໂນດໃໝ່ໃນລະດັບຖັດໄປໃຫ້ກັບໂຄງສ້າງຕົ້ນໄມ້

3. ຖ້າມີ ໃຫ້ກະທຳດັ່ງຕໍ່ໄປນີ້

- ຕັ້ງໃຫ້ໂນດທີ່ຈະສ້າງໃໝ່ຊີ້ໄປທີ່ໂນດທີ່ມີຢູ່ແລ້ວ ແລະເອົາໂນດທີ່ຈະເກີດຖິ້ມໄປ

2.1.5 ການສະແດງຄວາມຮູ້

ເປັນຂະບວນການໃນການສະແດງຄວາມໝາຍທີ່ປາກົດຢູ່ໃນແຕ່ລະໂນດວ່າຈະມີ ວິທີການ ແນວໃດ, ສຳລັບເລື່ອງກ່ຽວກັບການສະແດງຄວາມຮູ້ຈະໄດ້ມີການອະທິບາຍລະອຽດໃນບົດຕໍ່ໄປ. ໃນ ກໍລະນີຂອງໂຖນ້ຳດັ່ງທີ່ໄດ້ກ່າວໄວ້ໃນບົດນີ້ ອາໄສວິທີການສະແດງຄວາມໝາຍໂດຍໃຊ້ຊຸດຂອງຕົວເລກ ໂດຍຕົວທຳອິດສະແດງການສະແດງຄວາມຮູ້ນອກຈາກການກຳນົດວິທີການທີ່ຈະສະແດງຄ່າຂອງໂນດແລ້ວ ການສະແດງຄວາມຮູ້ຍັງຈະຕ້ອງປະກອບດ້ວຍສ່ວນທີ່ສະແດງຄວາມສຳພັນຂອງເງື່ອນໄຂການປ່ຽນແປງ ສະຖານະດັ່ງນີ້:

(X<4,Y) -> (4,Y)

ສົມຜົນຂ້າງຕົ້ນເປັນການສະແດງຄວາມສຳພັນຂອງການປ່ຽນສະຖານະ ເຊິ່ງອະທິບາຍໄດ້ວ່າ ຖ້າ (X<4,Y) ຄື ຄ່າຂອງ X ນ້ອຍກວ່າ 4 ແລະຄ່າຂອງ Y ເປັນຄ່າໃດ ๆ ເປັນຈິງ ເຮົາສາມາດທຳໃຫ້ຄ່າ ຂອງ X ເປັນ 4 ແລະຄ່າ Y ຄົງທີ່ໄດ້.

ວິທີດັ່ງກ່າວເປັນການສະແດງຄວາມຮູ້ແບບງ່າຍໆ ແຕ່ໃນກໍລະນີທີ່ຄວາມຮູ້ມີຄວາມຊັບຊ້ອນຂຶ້ນມີ ເລື່ອງຫຼາຍຢ່າງທີ່ຈະຕ້ອງພິຈາລະນາ ເຊັ່ນ: ໃນກໍລະນີທີ່ຄວາມຮູ້ບໍ່ແມ່ນຕົວເລກ ແຕ່ແມ່ນອອບເຈັກ (object) ແລະ ຄວາມຈິງ (fact) ທີ່ມີຄວາມສຳພັນກັນ ເຊັ່ນ: ຄວາມຮູ້ 'Plant is on the table' ຈະ ມີອອບເຈັກ 2 ຕົວຄື Plant ແລະ Table ທີ່ມີ on ສະແດງເຖິງຄວາມສຳພັນ ການສະແດງຄວາມຮູ້ແບບ ນີ້ຈະມີການກ່າວເຖິງຕໍ່ໄປຢ່າງລະອຽດ ສຳລັບຕົວຢ່າງຂອງການສະແດງຄວາມຮູ້ແບບນີ້ມີຕົວຢ່າງດັ່ງຕໍ່ ໄປນີ້:

ON(plant,table) : plant is on the table

IN(table,room) : table is in the room

UNDER(table,window) : table is under the window

ຢ່າງໃດກໍ່ຕາມການສະແດງຄວາມຮູ້ມີສິ່ງທີ່ຄວນຈະຄຳນຶງເຖິງດັ່ງຕໍ່ໄປນີ້:

1. ຄວາມຮູ້ທັງໝົດຈະສາມາດລວມເປັນຄວາມຮູ້ດຽວກັນໄດ້ແນວໃດ ເຊັ່ນຫາກວ່າເຮົາກຳລັງອະທິບາຍເຖິງລັກສະນະຂອງຫ້ອງຫ້ອງໜຶ່ງທີ່ບອກວ່າ ຫ້ອງນີ້ຕັ້ງໂຕະໄວ້ກ້ອງປ້ອງຢັ້ງມ 'table is under the window' ແລ້ວອີກໜຶ່ງເມື່ອມີການປ່ຽນຖານຄວາມຮູ້ວ່າ CENTER(table,room) ໃນລະບົບການສະແດງຄວາມຮູ້ຈະມີວິທີການແນວໃດທີ່ຈະ

ເຮັດໃຫ້ຮູ້ວ່າ UNDER(table,window) ໃຊ້ບໍ່ໄດ້ແລ້ວເພາະເມື່ອໂຕະມາຢູ່ກາງຫ້ອງກໍເປັນໄປບໍ່ໄດ້ທີ່ໂຕະໂຕດຽວກັນຈະຢູ່ກ້ອງປ້ອງຢັ້ງມ.

2. ການຈັດລຳດັບແນວໃດເຮັດໃຫ້ການຄົ້ນຫາເຮັດໄດ້ງ່າຍ ເຊັ່ນ ຖ້າຈະເຕີມABOVE(ceiling, floor) ເຂົ້າໄປໃນຖານຄວາມຮູ້ ຈະໃສ່ບ່ອນໃດທີ່ຈະໄດ້ບໍ່ຕ້ອງບອກທຸກເທື່ອເມື່ອມີການກ່າວອ້າງເຖິງເລື່ອງຂອງຫ້ອງເພາະ ' ເພດານຢູ່ເໜືອພື້ນ' ນີ້ເປັນຄວາມຈິງທົ່ວໄປໃນເລື່ອງທີ່ກ່ຽວກັບຫ້ອງ.

ຂະບວນການດັ່ງກ່າວມາຂ້າງເທິງທັງໝົດ, ຄວາມຈິງແລ້ວຈະເປັນເລື່ອງທີ່ກ່ຽວກັບການສະແດງຄວາມຮູ້ໂດຍເຟມ(frame)ເຊິ່ງຈະໄດ້ມີການອະທິບາຍລະອຽດ ໃນບົດທີ 3

2.1.6 ຂະບວນການໃນການເລືອກກົດເກນ

ໂດຍປົກກະຕິແລ້ວວິທີໃນການເລືອກກົດເກນສາມາດເຮັດໄດ້ໂດຍວິທີການປຽບທຽບ(matching) ເຊິ່ງເຮັດໄດ້ໂດຍການນຳສະຖານະປັດຈຸບັນ(current state) ໄປປຽບທຽບກັບ ເງື່ອນໄຂຂອງກົດ ຂະບວນການຂອງການປຽບທຽບທີ່ນິຍົມໃຊ້ກັນຫຼາຍມີຢູ່ 2 ວິທີຄື

1. ການເຮັດດັດນີ (Indexing)

ຫຼັກການຂອງການແກ້ປັນຫາແບບນີ້ຄືໃຫ້ຫາກົດທຸກຂໍ້ທີ່ເງື່ອນໄຂກົງກັບເງື່ອນໄຂ ທີ່ວາງເອົາໄວ້ໃນສຖານະປັດຈຸບັນແລ້ວດຶງເອົາກົດທຸກຂໍ້ນັ້ນອອກມາ ເພື່ອຈະທຳການປຽບທຽບ ຖ້າຫາກວ່າໄດ້ຄຳຕອບການແກ້ປັນຫາກໍສິ້ນສຸດ ຖ້າຫາກວ່າບໍ່ພົບຄຳຕອບ ໃຫ້ທຳການປຽບທຽບໃໝ່ຈົນພົບຄຳຕອບ.

ຖ້າເບິ່ງຈາກເລື່ອງຂອງໂຖນ້ຳເຮົາສາມາດອະທິບາຍກົດໃນລັກສະນະຂອງເງື່ອນໄຂແລະຂໍ້ສະຫຼຸບໄດ້ດັ່ງຮູບທີ 2.4 ໃນສ່ວນຂອງເງື່ອນໄຂເປັນສ່ວນທີ່ບອກເງື່ອນໄຂຂອງການໃຫ້ກົດ ແລະໃນສ່ວນຂອງການສ້າງເປັນສ່ວນທີ່ບອກເຖິງສະຖານະໃໝ່ທີ່ຈະເປັນຂອງສະຖານະຕ໊ອຢ່າງ ເບິ່ງທີ່ກົດຂໍ້ທີ່ 1 ຈາກຕາຕະລາງ ໃນຮູບທີ 2.9 ຈະສາມາດເຂົ້າໃຈໄດ້ວ່າຖ້າຫາກໃນສະຖານະປັດຈຸບັນ 'ໂຖນ້ຳທີ 1 ມີນ້ຳນ້ອຍກວ່າ 4 ລິດ' ແລ້ວໃຫ້ 'ເຕີມນ້ຳລົງໃນໂຖນ້ຳທີ 1 ຈົນເຕັມ' ຜົນທີ່ໄດ້ຈະເປັນ 'ໂຖນ້ຳທຳອິດມີນ້ຳ 4 ລິດ ແລະໂຖທີສອງມີນ້ຳຢູ່ເທົ່າເດີມ(4,Y)'

ຖ້າຫາກຈະພິຈາລະນາຈາກສະຖານະເລີ່ມຕົ້ນທີ່ມີນ້ຳໃນໂຖນ້ຳທັງສອງເປັນ 0 ຫຼື (0,0) ແລ້ວຈະພິຈາລະນາຈາກເງື່ອນໄຂຕາມຕາຕະລາງໃນຮູບທີ 2.4 ຈະເຫັນໄດ້ວ່າ ມີກົດຂໍ້ທີ 1 ແລະ 2 ເທົ່ານັ້ນທີ່ມີເງື່ອນໄຂທີ່ກົງກັນ.

ເມື່ອນຳເອົາກົດຂໍ້ທີ 1 ມາໃຊ້ ສະຖານະຈະປ່ຽນຈາກ(0,0) ເປັນ(4,0)

ເມື່ອນຳເອົາກົດຂໍ້ທີ 2 ມາໃຊ້ ສະຖານະຈະປ່ຽນຈາກ(0,0) ເປັນ(0,3)

ຈາກສະຖານະ (4,0) ມີກົດຂໍ້ 2, 5 ແລະ 8 ທີ່ມີເງື່ອນໄຂກົງກັນ ແລະສະຖານະຕ່າງໆ ຈະປ່ຽນໄປດັ່ງນີ້

(4,0) -> (4,3) ໂດຍການໃຊ້ກົດຂໍ້ 2

(4,0) -> (0,0) ໂດຍການໃຊ້ກົດຂໍ້ 3

(4,0) -> (1,3) ໂດຍການໃຊ້ກົດຂໍ້ 6

ຈາກສະຖານະ (0,3) ມີກົດຂໍ້ 1, 6 ແລະ 9 ທີ່ມີເງື່ອນໄຂກົງກັນ ສະຖານະຕ່າງໆ ຈະປ່ຽນໄປເປັນດັ່ງນີ້

(0,3) -> (4,3) ໂດຍກົດຂໍ້ 1

(0,3) -> (0,0) ໂດຍກົດຂໍ້ 4

(0,3) -> (3,0) ໂດຍກົດຂໍ້ 7

ຖ້າເບິ່ງຈາກໂຄງສ້າງຕົ້ນໄມ້ທີ່ສ້າງຂຶ້ນ ຈະເຫັນວ່າມີຢູ່ຫຼາຍສະຖານະທີ່ບໍ່ມີການສ້າງສະຖານະໃໝ່ ເນື່ອງຈາກວ່າສະຖານະເຫຼົ່ານັ້ນໄດ້ເກີດຂຶ້ນມາແລ້ວກ່ອນໜ້ານັ້ນ ໃນລະດັບທີ່ສູງກວ່າ ເປັນແນວນີ້ເພາະຖ້າຫາກວ່ານຳສະຖານະເຫຼົ່ານັ້ນມາສ້າງສະຖານະໃໝ່ ຈະເຮັດໃຫ້ເກີດການວົນລູບ (loop) ຂຶ້ນ ເຊິ່ງກໍໝາຍຄວາມວ່າບໍ່ມີທາງຈະພົບຄຳຕອບ

ນອກຈາກປັນຫານີ້ແລ້ວ ໃນການແກ້ປັນຫານີ້ມີປັນຫາອື່ນທີ່ຕ້ອງພິຈາລະນາ

- ຕ້ອງໃຊ້ກົດຫຼາຍຂໍ້ເພື່ອອະທິບາຍແຕ່ລະເງື່ອນໄຂ
- ການຄົ້ນຫາໃຊ້ເວລາດົນ

2. ການຈັບຄູ່ກັບຕົວປ່ຽນ (Matching with variable)

ວິທີນີ້ຂຶ້ນຢູ່ກັບການປຽບທຽບກົດແລະຄວາມຈິງ ເພື່ອຄວາມເຂົ້າໃຈໃຫ້ເບິ່ງຕົວຢ່າງຕໍ່ໄປນີ້:

ຄວາມເປັນຈິງ: ແດງໃຫຍ່ເປັນພໍ່ຂອງແດງ

ຄວາມເປັນຈິງ: ແດງເປັນພໍ່ຂອງແດງນ້ອຍ

ຈາກຄວາມເປັນຈິງດັ່ງກ່າວ ຖ້າຈະຖາມວ່າ `ແດງໃຫຍ່ເປັນຫຍັງກັບແດງນ້ອຍ' ຄອມພິວເຕີຈະບອກເຖິງຄວາມສຳພັນນີ້ບໍ່ໄດ້ ສິ່ງທີ່ຄອມພິວເຕີຈະບອກເຖິງຄວາມສຳພັນນີ້ໄດ້ ຈະຕ້ອງມີກົດບາງຢ່າງທີ່ກ່າວເຖິງເລື່ອງດັ່ງກ່າວ ເຊັ່ນ:

rule: ຖ້າ X ເປັນພໍ່ຂອງ Y

ແລະ Y ເປັນພໍ່ຂອງ Z

ດັ່ງນັ້ນ X ເປັນປູ່ຂອງ Z

ເມື່ອມີຄວາມຈິງແລະກົດດັ່ງທີ່ໄດ້ສະແດງແລ້ວ, ຄວາມເປັນໄປໄດ້ຂອງການຕອບຄຳຖາມຈະເຫັນແຈ້ງຂຶ້ນ ໂດຍການແທນຄ່າ X ຄືແດງໃຫຍ່ ແລະ Y ຄືແດງ ແລະ Z ຄືແດງນ້ອຍ ເຮົາກໍຈະໄດ້ຄວາມສຳພັນຂອງແດງໃຫຍ່ແລະແດງນ້ອຍວ່າ ຄືປູ່ຕາມກົດ ເຊິ່ງລັກສະນະແບບນີ້ເອງທີ່ເຮົາເອີ້ນວ່າ *ການຫາຂໍ້ສະຫຼຸບ*

ການຫາຂໍ້ສະຫຼຸບດັ່ງທີ່ກ່າວມາມີກົດພຽງຂໍ້ດຽວເທົ່ານັ້ນ ເຊິ່ງເຮັດໄດ້ງ່າຍແຕ່ໃນກໍລະນີທີ່ມີກົດຫຼາຍຂຶ້ນ ເຊັ່ນ ຖ້າຫາກເພີ່ມກົດເຂົ້າໄປອີກ 1 ຂໍ້ດັ່ງນີ້

rule 1: ຖ້າ X ເປັນອ້າຍຂອງ Y ແລະ

Y ເປັນອ້າຍຂອງ Z

ດັ່ງນັ້ນ X ເປັນອ້າຍຂອງ Z

ເມື່ອເປັນແນວນີ້ ປັນຫາກໍຈະເກີດຂຶ້ນຕາມມາວ່າເຮົາຈະເລືອກກົດເກນຂໍ້ໃດໃນການຫາຂໍ້ສະຫຼຸບລະຫວ່າງ rule ແລະ rule1 ໃນນີ້ຫາກສຳຫຼວດເບິ່ງຈາກ fact ທີ່ມີຢູ່ ຈະເຫັນໄດ້ແຈ້ງວ່າຈະຕ້ອງເລືອກ rule ເພາະຄວາມສຳພັນຂອງ `ແດງໃຫຍ່' `ແດງ' ແລະ `ແດງນ້ອຍ' ເປັນ `ພໍ່' ບໍ່ແມ່ນ `ອ້າຍ ' ດັ່ງນັ້ນໃນຫາຂໍ້ສະຫຼຸບສ່ວນທີ່ເຮັດໜ້າ ທີ່ໃນ
ຫາຂໍ້ສະຫຼຸບຈະຕ້ອງກວດສອບຄວາມສຳພັນ(relation 'ພໍ່' ແລະ 'ອ້າຍ') ຂອງ ແອັດທິບິວ (attribute 'ແດງໃຫຍ່' 'ແດງ' ແລະ 'ແດງນ້ອຍ') ວ່າກ່ຽວພັນກັບກົດຂໍ້ໃດ ຈຶ່ງນຳກົດຂໍ້ນັ້ນໄປຫາຂໍ້ສະຫຼຸບ

ວິທີການແກ້ປັນຫາແບບນີ້ ຈະຕ້ອງອາໄສວິທີການສະແດງຄວາມຮູ້ທີ່ແຕກຕ່າງຈາກການສະແດງຄວາມຮູ້ໃນເລື່ອງໂຖນ້ຳດັ່ງທີ່ສະແດງຜ່ານມາ ເປັນການແກ້ປັນຫາຂອງຄວາມຮູ້ທີ່ຖືກສະແດງໂດຍກົດ ການສະແດງຄວາມຮູ້ອີກແບບໜຶ່ງທີ່ສາມາດໃຊ້ໄດ້ດີກັບວິທີການນີ້ຄື ແບບ Predicate Logic

ຕົວຢ່າງຂອງການສະແດງຄວາມຮູ້ດັ່ງກ່າວ

Facts:

SON (John,Mary) :John ເປັນລູກຊາຍຂອງ Mary

SON (Bill,John) :Bill ເປັນລູກຊາຍຂອງ John

SON (Tom,Bill) :Tom ເປັນລູກຊາຍຂອງ Bill

SON (Joe,Bill) :Joe ເປັນລູກຊາຍຂອງ Bill

DAUGHTER (Sue,John) :Sue ເປັນລູກສາວຂອງ John

1. SON(x,y)^SON(y,z) -> GRANDSON(x,z) :
ຖ້າ x ເປັນລູກຊາຍຂອງ y ແລະ y ເປັນລູກຊາຍຂອງ z
ດັ່ງນັ້ນ x ຈະເປັນຫຼານຊາຍຂອງ z

2. DAUGHTER(x,y)^SON(y,z) -> GRANDAUGHTER (x,z) :

ຖ້າ x ເປັນລູກສາວຂອງ y ແລະ

y ເປັນລູກສາວຂອງ z

ດັ່ງນັ້ນ x ຈະເປັນຫຼານສາວຂອງ z

3. SON(x,y)^DAUGHTER(y,z) -> GRANDSON(x,z) :

ຖ້າ x ເປັນລູກຊາຍຂອງ y ແລະ

y ເປັນລູກສາວຂອງ z

ດັ່ງນັ້ນ x ຈະເປັນຫຼານສາວຂອງ z

ໃນການເລືອກກົດເກນ ນອກຈາກການພິຈາລະນາວ່າເຮົາຈະມີວິທີການເລືອກກົດເກນຂໍ້ໃດແລ້ວການເລືອກກົດເກນຍັງຈະຕ້ອງສົນໃຈການກຳນົດລຳດັບຂອງການພິຈາລະນາກົດໂດຍ ຄືການພິຈາລະນາວ່າເຮົາຈະພິຈາລະນາກົດຂໍ້ໃດກ່ອນ ຫຼື ຫຼັງເພາະລຳດັບຂອງການພິຈາລະນາກົດ ຈະມີຜົນຕໍ່ການແກ້ປັນຫາ, ໂດຍປົກກະຕິແລ້ວການກຳນົດລຳດັບຂອງການຄົ້ນຫາ ມີ 2 ຢ່າງຄື

- ການຫາເຫດຜົນແບບໄປໜ້າ(Forward Reasoning) ແລະ
- ການຫາເຫດຜົນແບບຍ້ອນກັບຫຼັງ(Backward Reasoning)

ການຫາເຫດຜົນແບບໄປໜ້າແລະແບບຍ້ອນກັບຫຼັງເປັນຂະບວນການຂອງການກຳນົດທິດທາງສຳລັບການຄົ້ນຫາ ແລະການເລືອກກົດເກນຂໍ້ທີ່ເໝາະສົມ ເພື່ອທີ່ຈະກຳນົດທິດທາງວ່າໃນຫາຂໍ້ສະຫຼຸບ ຈະເລີ່ມຕົ້ນການຫາຄຳຕອບຈາກສ່ວນໃດຂອງຂອບເຂດປັນຫາ(Problem Space) ແລະຈັດກຽມກົດ (ຫຼື ຖານຄວາມຮູ້:Knowledge Base)ໃຫ້ເໝາະກັບການຄົ້ນຫາຄຳຕອບແບບນັ້ນ. ໂດຍປົກກະຕິແລ້ວຈະອາໄສໂຄງສ້າງຂໍ້ມູນແບບຕົ້ນໄມ້(Tree Structure) ແລະ / ຫຼືແບບກຣາບ (Graph) ເປັນຮູບແບບຫຼັກ.

ການຫາເຫດຜົນແບບໄປໜ້າ ຈະເລີ່ມຕົ້ນຈາກສະຖານະເລີ່ມຕົ້ນນີ້ມາເປັນຣາກ (Root) ຂອງໂຄງສ້າງຕົ້ນໄມ້ ແລະສ້າງໂນດລູກ (Successor) ໃນລະດັບຖັດໄປ ໂດຍການຫາກົດທຸກຂໍ້ ເຊິ່ງທາງດ້ານຊ້າຍຂອງກົດທີ່ກົງກັບໂນດຣາກ ແລະໃຊ້ທາງດ້ານຂວາຂອງກົດສ້າງເປັນລູກຂອງໂນດນັ້ນ ແລະສ້າງລະດັບຕໍ່ໄປໂດຍການເອົາໂນດລູກນັ້ນມາທຳໂດຍວິທີດຽວກັນຈົນຄົບທຸກໂນດຈົນໝົດໃນລະດັບນັ້ນ ແລະໃຫ້ເຮັດແບບນີ້ໄປເລື້ອງຈົນຄົບກົດທຸກຂໍ້. ໃນການຫາເຫດຜົນແບບນີ້ບາງເທື່ອຈະເອີ້ນວ່າ D*ata-Driven Reasoning* ເພາະການຄົ້ນຫາເລີ່ມຈາກຂໍ້ມູນພາຍໃນກົດ.

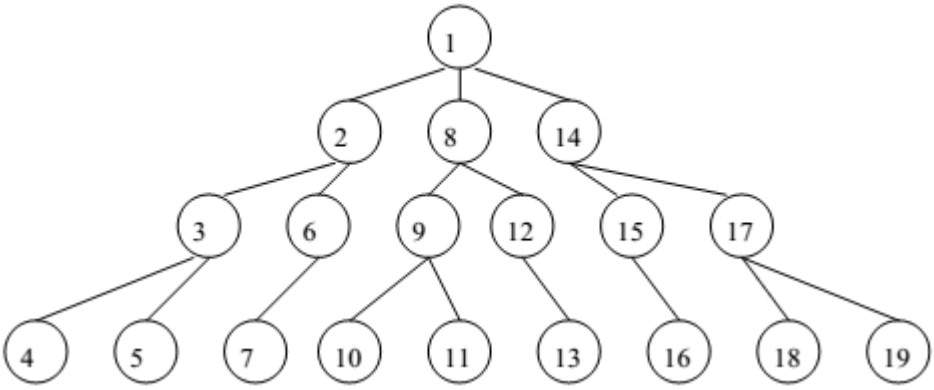
ການຫາເຫດຜົນແບບຍ້ອນກັບຫຼັງ ຈະເລີ່ມຕົ້ນຈາກສຖານະເປົ້າໝາຍ ໂດຍການນຳເອົາເປົ້າໝາຍນັ້ນມາສ້າງເປັນຣາກຂອງໂຄງສ້າງຕົ້ນໄມ້ ແລະສ້າງລູກໃນລະດັບຖັດໄປ ໂດຍການຫາກົດທຸກຂໍ້ ເຊິ່ງທາງດ້ານຊ້າຍຂອງກົດກົງກັບໂນດຣາກ ແລະໃຊ້ທາງດ້ານຂວາຂອງກົດສ້າງເປັນລູກຂອງໂນດນັ້ນ ແລະສ້າງລະດັບຕໍ່ ໄປໂດຍການເອົາລູກໂນດນັ້ນມາສ້າງວິທີດຽວກັນຈົນຄົບທຸກໂນດ ຈົນໝົດໃນລະດັບນັ້ນ ແລະໃຫ້ເຮັດແບບນີ້ໄປເລື້ອຍໆ ຈົນຄົບກົດທຸກຂໍ້ ໃນການຫາເຫດຜົນແບບນີ້ບາງເທື່ອຈະເອີ້ນວ່າ: Goal-Driven Reasoning ເພາະການຫາເຫດຜົນເລີ່ມຕົ້ນຈາກເປົ້າໝາຍ (Goal)

2.2. ການຄົ້ນຫາແບບງົມມືດ (Blind Search)

ການຄົ້ນຫາແບບງົມມືດ(Blind search) ເປັນການຄົ້ນຫາແບບທີ່ເດີນທາງຈາກໂນດໜຶ່ງໄປຍັງອີກໂນດໜຶ່ງ ໂດຍອາໄສທິດທາງເປັນຕົວກຳນົດການຄົ້ນຫາ ບໍ່ຕ້ອງມີຂໍ້ມູນຫຍັງມາຊ່ວຍໃນການຕັດສິນໃຈວ່າຈະເດີນທາງຕໍ່ໄປແບບໃດ ຫຼື ເວົ້າອີກແບບໜຶ່ງ ຄືການຈະເລືອກເອົາຂໍ້ມູນໃດມາຊ່ວຍໃນການຄົ້ນຫາຕໍ່ໄປ ບໍ່ຕ້ອງອາໄສຂໍ້ມູນໃດໆໝົດ ນອກຈາກທິດທາງ ຕົວຢ່າງຂອງການຄົ້ນຫາຂໍ້ມູນແບບນີ້ຄື ການຄົ້ນຫາແບບເລິກກ່ອນ(Depth First Search) ແລະ ການຄົ້ນຫາແບບກວ້າງກອນ (Breadth First Search)

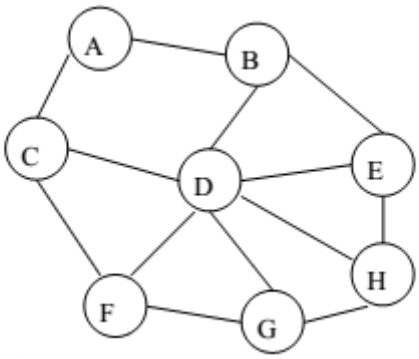
2.2.1 ການຄົ້ນຫາແບບລ່ວງເລີກກ່ອນ

ການຄົ້ນຫາແບບເລີກກ່ອນເປັນການຄົ້ນຫາທີ່ນຳນິດທິດທາງຈາກຮູບຂອງໂຄງສ້າງຕົ້ນໄມ້ ທີ່ເລີ່ມຕົ້ນຈາກໂນດຮາກ (Root node) ທີ່ຢູ່ເທິງສຸດແລ້ວລົງມາໃຫ້ເລີກທີ່ສຸດ ເມື່ອຮອດໂນດລຸ່ມສຸດ(Terminal Node) ໃຫ້ກັບຂຶ້ນມາທີ່ຈຸດສູງສຸດຂອງກິ່ງດ່ຽວກັນທີ່ມີກິ່ງແຍກ ແລະ ຍັງບໍ່ໄດ້ແລ້ວເລີ່ມລົງໄປຈົນຮອດໂນດເລີກສຸດອີກ ເຮັດແບບນີ້ສະຫຼັບໄປເລື້ອຍໆຈົນພົບໂນດທີ່ຕ້ອງການຫາ ຫຼື ກວດສອບຄົບທຸກໂນດ ແລ້ວຕາມຮູບທີ 2.10 ການຄົ້ນຫາແບບເລີກກ່ອນຈະມີລຳດັບການໄປຕາມໂນດດັ່ງຕົວເລກທີ່ລະບຸໄວ້ໃນແຕ່ລະໂນດ



ຮູບທີ 2.10 ລຳດັບການຄົ້ນຫາແບບລວງເລີກກ່ອນແບບໂຄງສ້າງຕົ້ນໄມ້

ດັ່ງທີ່ໄດ້ກ່າວມາແລ້ວວ່າໂຄງສ້າງຂໍ້ມູນທີ່ໃຊ້ສຳລັບການຄົ້ນຫານີ້ສາມາດໃຊ້ກັບໂຄງສ້າງກຣາບໄດ້ ໂດຍອາໄສຫຼັກການດຽວກັນ, ແຕ່ສຳລັບການເດີນທາງເທິງກຣາບນັ້ນຈະບໍ່ມີໂນດເລີກສຸດ, ດັ່ງນັ້ນການເດີນທາງເທິງກຣາບຈະຕ້ອງປັບວິທີການເປັນດັ່ງນີ້: ໂດຍເລີ່ມຈາກໂນດເລີ່ມຕົ້ນ ຈາກນັ້ນໃຫ້ນຳໂນດທີ່ຢູ່ຕິດກັບໂນດທີ່ກຳລັງກວດສອບຢູ່ (ທີ່ຍັງບໍ່ໄດ້ສ້າງການກວດສອບ ແລະ ຍັງບໍ່ໄດ້ຢູ່ໃນສະແຕ່ກມາໃສ່ສະແຕ່ກ) ມາເກັບໄວ້ໃນສແຕັກ, ເມື່ອກວດສອບໂນດນັ້ນສຳເລັດໃຫ້ປ່ອບ (Pop) ຕົວເທິງສຸດຂອງໂນດອອກມາສ້າງການກວດສອບ, ແລ້ວນຳໂນດຂ້າງຄຽງທັງໝົດທີ່ຍັງບໍ່ໄດ້ກວດສອບມາຕໍ່ທ້າຍສະແຕ່ກ ແລ້ວປ່ອບຕົວເທິງສຸດອອກມາກວດສອບ. ເຮັດແບບນີ້ເລື້ອຍໆໆຈົນກ່ວາພົບໂນດທີ່ຕ້ອງການ ຫຼື ກວດສອບຄົບທຸກໂນດ



ຮູບທີ 2.11 ໂຄງສ້າງຂໍ້ມູນແບບກຣາບ

ການກວດສອບຈະເລີ່ມຕົ້ນທີ່ A ແລະ ນຳໂນດຂ້າງຄຽງ B ແລະ C ມາເກັບໄວ້ໃນສະແຕ໋ກເມື່ອ ກວດສອບ A ສຳເລັດປ່ອບຂໍ້ມູນຈາກສະແຕ໋ກອອກມາໄດ້ C ສ້າງການກວດສອບ Cແລະນຳໂນດ ຂ້າງຄຽງກັບ C ທີ່ຍັງບໍ່ໄດ້ສ້າງການກວດສອບແລະຍັງບໍ່ໄດ້ຢູ່ໃນສະແຕ໋ກມາໃສ່ສະແຕ໋ກ D ແລະ F ພຸສ (Push)ໃສ່ສະແຕ໋ກດັ່ງນັ້ນໃນສະແຕ໋ກຕອນນີ້ມີ B D F ຢູ່ ເມື່ອກວດສອບ C ສຳເລັດ ປ່ອບ F ອອກມາ ສ້າງການກວດສອບແລ້ວນຳໂນດຂ້າງຄຽງທີ່ຍັງບໍ່ໄດ້ກວດສອບແລະຍັງບໍ່ໄດ້ຢູ່ໃນສະແຕ໋ກມາໃສ່ສະແຕ໋ກ ເຊິ່ງກໍຄື G ດັ່ງນັ້ນຂໍ້ມູນໃນສະແຕ໋ກຈະເປັນ B D G ເຮັດແບບນີ້ໄປເລື້ອຍໆ ໆ ຈົນຈົບການເຮັດວຽກກໍຈະ ໄດ້ລຳດັບການກວດສອບຄື (A C F G H E D B) ຕາມຕາຕະລາງ 1 ດັ່ງຕໍ່ໄປນີ້:

ໂນດທີ່ສຳຫຼວດ	ສະແຕ໋ກ
A	B C
C	B D F
F	B D G
G	B D H
H	B D E
E	B D
D	B
B	

ຕາຕະລາງທີ 2.2 ລຳດັບການຄົ້ນຫາແບບລວງເລິກກ່ອນ

ໃນການຄົ້ນຫາຂໍ້ມູນແບບນີ້ເທິງໂຄງສ້າງຂອງກຣາບ ມີສິ່ງທີ່ຄວນສັງເກດຄື ໂນດທີ່ເລີ່ມຕົ້ນການກວດ ສອບຈະຕ້ອງມີການກຳນົດມາໃຫ້ວ່າໂນດໃດເປັນໂນດເລີ່ມຕົ້ນ ແລະຂໍ້ ສັງເກດອີກຢ່າງໜຶ່ງຄືວິທີການຄົ້ນ ຫາແບບເລິກກ່ອນທີ່ໃຊ້ສຳລັບໂຄງສ້າງຂໍ້ມູນແບບກຣາບ ສາມາດໃຊ້ກັບໂຄງສ້າງຂໍ້ມູນແບບຕົ້ນໄມ້ໄດ້ ດ້ວຍ

- **ອານກໍຣິດທຶມ ການຄົ້ນຫາຂໍ້ມູນແບບເລິກກ່ອນ**

ໃຫ້ສະຖານະ 1 ໝາຍເຖິງໂນດທີ່ຍັງບໍ່ກວດສອບ, ສະຖານະ 2 ໝາຍເຖິງໂນດທີ່ຢູ່ໃນ STACK ແລະ ສະຖານະ 3
ໝາຍເຖິງໂນດທີ່ສ້າງການກວດສອບແລ້ວ

1) ທຳໃຫ້ໂນດທຸກໂນດມີສະຖານະເປັນ 1 ແລະນຳໂນດເລີ່ມຕົ້ນໄວ້ໃນ STACK ປ່ຽນສະຖານະ ເປັນ 2
2) ນຳໂນດເທິງສຸດໃນ STACK ອອກມາກວດສອບ ແລະປ່ຽນສະຖານະເປັນ 3

-ຖ້າໂນດທີ່ສ້າງການກວດສອບຢູ່ຄືໂນດເປົ້າໝາຍລາຍງານໂນດທີ່ກວດສອບຄືໂນດເປົ້າໝາຍ ແລະໃຫ້ຂ້າມໄປທີ່ຂັ້ນຕອນ 4

-ຖ້າໂນດທີ່ກວດສອບບໍ່ໃຊ້ໂນດເປົ້າໝາຍ ແລະຈຳນວນໂນດໃນ STACK ມີຫຼາຍກວ່າ 0 ໃຫ້

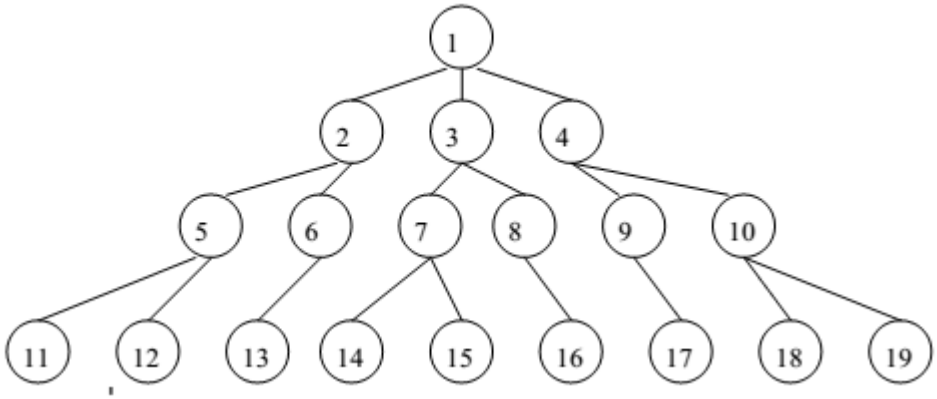
ນຳໂນດຂ້າງຄຽງທີ່ມີສະຖານະເປັນ 1 ທັງໝົດ(ຖ້າມີ) ປ່ຽນສະຖານະເປັນ 2 ແລ້ວນຳໃສ່ໄວ້ໃນ STACK ແລະ ກັບໄປເຮັດຂໍ້ທີ 2

3) ລາຍງານການຄົ້ນຫາບໍ່ໄດ້
4) ການເຮັດວຽກສິ້ນສຸດ

ໝາຍເຫດ ລັກສະນະການໃສ່ຂໍ້ມູນເຂົ້າແລະການນຳຂໍ້ມູນອອກຂອງໂຄງສ້າງຂໍ້ມູນແບບສະແຕ໋ກ ຈະເປັນໃນລັກສະນະຂໍ້ມູນທີ່ນຳເຂົ້າກ່ອນ ຈະຖືກນຳມາໃຊ້ພາຍຫຼັງ

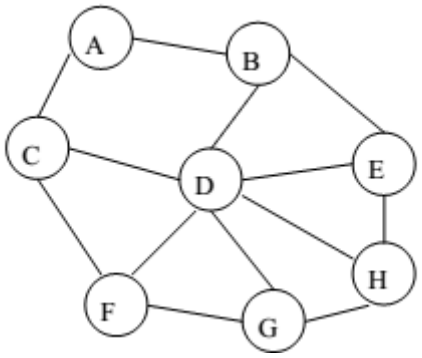
2.2.2. ການຄົ້ນຫາແບບລວງກວ້າງກ່ອນ

ການຄົ້ນຫາແບບລວງກວ້າງກ່ອນເປັນການກຳນົດທິດທາງການຄົ້ນຫາແບບເທື່ອລະລະດັບຂອງໂຄງສ້າງຕົ້ນໄມ້ໂດຍເລີ່ມຈາກໂນດຮາກ (ລະດັບທີ 0) ແລ້ວລົງມາລະດັບທີ 1 ຈາກຊ້າຍໄປຂວາ ເມື່ອສຳເລັດລະດັບທີ 1 ໄປລະດັບທີ 2 ຈາກຊ້າຍໄປຂວາເຊັ່ນກັນ ເຮັດແບບນີ້ເລື້ອຍໆ ຈົນພົບໂນດທີ່ຕ້ອງການຕາມຮູບທີ 3 ລຳດັບການເດີນທາງຂອງໂນດເປັນໄປຕາມໝາຍເລກທີ່ລະບຸໄວ້ເທິງໂນດ



ຮູບທີ 2.12 ລຳດັບການຄົ້ນຫາແບບລວງກວ້າງກ່ອນແບບໂຄງສ້າງຕົ້ນໄມ້

ສຳລັບການຄົ້ນຫາແບບລວງກວ້າງກ່ອນເທິງໂຄງສ້າງຕົ້ນໄມ້ ຈະອາໄສໂຄງສ້າງຂໍ້ມູນແບບຄິວ (Queue)ມາຊ່ວຍ ແລະ ດ້ວຍວິທີການອັນດຽວກັບການຄົ້ນຫາແບບເລິກກ່ອນຄື ໃຫ້ເລີ່ມຕົ້ນກວດສອບທີ່ໂນດເລີ່ມຕົ້ນແລ້ວນຳໂນດຂ້າງຄຽງເກັບໄວ້ໃນຄິວ ເມື່ອກວດສອບໂນດເລີ່ມຕົ້ນສຳເລັດ ໃຫ້ນຳຂໍ້ມູນໃນຄິວອອກມາກວດສອບແລ້ວນຳໂນດຂ້າງຄຽງທີ່ຍັງບໍ່ໄດ້ກວດສອບແລະບໍ່ໄດ້ຢູ່ໃນຄິວໃສ່ຄິວໄວ້ ເຮັດແບບນີ້ໄປເລື້ອຍໆຈົນພົບໂນດທີ່ຕ້ອງການ ຫຼື ເມື່ອກວດສອບຄົບທຸກໂນດ.



ຮູບທີ 2.13 ໂຄງສ້າງຂໍ້ມູນແບບກຣາບ

ການກວດສອບເລີ່ມຕົ້ນທີ່ A ນຳໂນດຂ້າງຄຽງ B,Cໄວ້ໃນຄິວເມື່ອກວດສອບ A ສຳເລັດນຳ ຂໍ້ມູນໃນຄິວຄື B ອອກມາກວດສອບແລ້ວນຳຂໍ້ມູນຂ້າງຄຽງຄື D, E ໃສ່ຄິວ ຕອນນີ້ຄິວຈະມີ B D E ຢູ່ ແລ້ວນຳ B ອອກມາກວດສອບ. ເຮັດແບບນີ້ເລື້ອຍໆຈະໄດ້ລຳດັບການກວດສອບຂໍ້ມູນຄື (A B C D E F G H)ຕາມຕາຕະລາງທີ 2.3

ໂນດທີ່ສຳຫຼວດ	ຄິວ
A	B C
B	C D E
C	D E F
D	E F G H
E	F G H
F	G H
G	H
H	

ຕາຕະລາງທີ 2.3 ລຳດັບການຄົ້ນຫາແບບລວງກວ້າງກ່ອນ

ຄືກັບການຄົ້ນຫາແບບເລິກກ່ອນ ການຄົ້ນຫາແບບລວງກວ້າງກ່ອນໂດຍໃຊ້ໂຄງສ້າງຂໍ້ມູນຄິວ ມາຊ່ວຍຕ້ອງມີການກຳນົດໂນດເລີ່ມຕົ້ນ ແລະວິທີການນີ້ສາມາດໃຊ້ໄດ້ກັບຂໍ້ມູນເທິງໂຄງສ້າງແບບຕົ້ນໄມ້

- **ອານກໍຣິດທຶມ ການຄົ້ນຫາຂໍ້ມູນແບບລວງກວ້າງກ່ອນ**

ໃຫ້ສະຖານະ 1 ໝາຍເຖິງໂນດທີ່ຍັງບໍ່ກວດສອບ, ສະຖານະ 2 ໝາຍເຖິງໂນດທີ່ຢູ່ໃນ QUEUE ແລະ ສະຖານະ 3 ໝາຍເຖິງໂນດທີ່ສ້າງການກວດສອບແລ້ວ

1) ທຳໃຫ້ໂນດທຸກໂນດມີສະຖານະເປັນ 1 ແລະນຳໂນດເລີ່ມຕົ້ນໄວ້ໃນ QUEUE ປ່ຽນສະຖານະ ເປັນ 2
2) ນຳໂນດທຳອິດໃນ QUEUE ອອກມາກວດສອບ ແລະປ່ຽນສະຖານະເປັນ 3
   - ຖ້າໂນດທີ່ສ້າງການກວດສອບຢູ່ຄືໂນດເປົ້າໝາຍ ລາຍງານໂນດທີ່ກວດສອບຄືໂນດເປົ້າໝ າຍ ແລະ ໃຫ້ຂ້າມໄປທີ່ຂັ້ນຕອນ 4
   - ຖ້າໂນດທີ່ກວດສອບບໍ່ແມ່ນໂນດເປົ້າໝາຍ ແລະຈຳນວນໂນດໃນ QUEUE ມີຫຼາຍກວ່າ 0 ໃຫ້ນຳໂນດຂ້າງຄຽງທີ່ມີສະຖານະເປັນ 1 ທັງໝົດ(ຖ້າມີ) ປ່ຽນສະຖານະເປັນ 2 ແລ້ວນຳໃສ່ ໄວ້ໃນ QUEUE ແລະ ກັບໄປຂໍ້ 2
3) ລາຍງານການຄົ້ນຫາທີ່ບໍ່ໄດ້
4) ການເຮັດວຽກສິ້ນສຸດ

   ໝາຍເຫດ: ລັກສະນະການໃສ່ຂໍ້ມູນເຂົ້າແລະການນຳຂໍ້ມູນອອກຂອງໂຄງສ້າງຂໍ້ມູນແບບຄິວ ຈະເປັນໃນລັກສະນະຂໍ້ມູນທີ່ນຳເຂົ້າກ່ອນ ຈະຖືກນຳມາໃຊ້ກ່ອນ

2.3. ການຄົ້ນຫາແບບຮິວຣິສະຕິກ (Heuristic Search)

ທາງດ້ານປັນຍາປະດິດ ການຄົ້ນຫາຄຳຕອບອາໄສວິທີການທາງຮິວຣິສະຕິກ (Heuristic Search) ມີຄວາມແຕກຕ່າງຈາກການຄົ້ນຫາຂໍ້ມູນແບບທຳມະດາຢູ່ທີ່ການຄົ້ນຫາຂໍ້ມູນແບບທຳມະດາຜູ້ທີ່ສ້າງການ ຄົ້ນຂໍ້ມູນຈະຕ້ອງກວດສອບຂໍ້ມູນເທື່ອລະຕົວ, ທຸກຕົວຈົນຄົບ ແຕ່ຮິວຣິສະຕິກຈະບໍ່ລົງໄປເບິ່ງຂໍ້ມູນທຸກຕົວ, ວິທີການນີ້ຈະເລືອກໄດ້ຄຳຕອບທີ່ເໝາະສົມໃຫ້ກັບການຄົ້ນຫາ ເຊິ່ງມີຂໍ້ດີຄື: ສາມາດສ້າງການຄົ້ນຫາຄຳ ຕອບຈາກ ຂໍ້ມູນທີ່ມີຂະໜາດໃຫຍ່ຫຼາຍໆ ໄດ້ ແຕ່ຂໍ້ເສຍຄືຄຳຕອບທີ່ໄດ້ເປັນພຽງຄຳຕອບທີ່ດີເທົ່ານັ້ນ, ແຕ່ ເນື່ອງຈາກວ່າປັນຫາໃນບາງລັກສະນະນັ້ນໃຫຍ່ຫຼາຍ ແລະ ເປັນໄປບໍ່ໄດ້ທີ່ຈະສ້າງການຄົ້ນຫດ້ວຍວິທີແບບ ທຳມະດາຂະບວນການຂອງຮິວຣິສະຕິກຈຶ່ງເປັນສິ່ງທີ່ຈຳເປັນ.

ໃນເລື່ອງຂອງຮິວຣິສະຕິກນັ້ນ ນອກຈາກຈະມີການຄົ້ນຫາແບບຮິວຣິສະຕິກແລ້ວ ຍັງມີອີກສິ່ງໜຶ່ງທີ່ ສຳຄັນຄື ຮິວຣິສະຕິກຟັງຊັນ(Heuristic Function) ເຊິ່ງໝາຍເຖິງຟັງຊັນທີ່ທຳໜ້າທີ່ໃນການວັດຂະໜາດ ຂອງຄວາມເປັນໄປໄດ້ໃນການແກ້ປັນຫາເຊິ່ງຈະສະແດງດ້ວຍຕົວເລກ

ວິທີການດັ່ງກ່າວຈະເຮັດໄດ້ໂດຍການພິຈາລະນາວິທີການ (Aspects) ຕ່າງໆ ທີ່ໃຊ້ໃນການແກ້ ປັນຫາທີ່ສະຖານະໜຶ່ງວ່າຈະສາມາດແກ້ປັນຫາໄດ້ຕາມທີ່ຕ້ອງການຫຼືບໍ່ ໂດຍກຳນົດເປັນນ້ຳໜັກທີ່ໃຫ້ກັບ ການແກ້ປັນຫາຂອງແຕ່ລະວິທີ ນ້ຳໜັກເຫຼົ່ານີ້ຈະຖືກສະແດງດ້ວຍຕົວເລກທີ່ລະບຸໄວ້ກັບໂນດຕ່າງໆ ໃນ ຂະບວນການຄົ້ນຫາ ແລະຄ່າເຫຼົ່ານີ້ຈະເປັນຕົວທີ່ໃຊ້ໃນການປະເມີນຄວາມເປັນໄປໄດ້ວ່າເສັ້ນທາງທີ່ຜ່ານ ໂນດນັ້ນຈະມີ ຄວາມເປັນໄປໄດ້ໃນການນຳໄປສູ່ຫົນທາງການແກ້ປັນຫາໄດ້ຫຼາຍໜ້ອຍຊ່ຳໃດ.

ຈຸດປະສົງແທ້ຈິງຂອງຮິວຣິສະຕິກ ຟັງຊັນກໍຄື ການລະບຸທິດທາງຂອງຂະບວນການຄົ້ນຫາ ເພື່ອໃຫ້ ຢູ່ໃນທິດທາງທີ່ໄດ້ປະໂຫຍດສູງສຸດ ໂດຍການບອກວ່າເຮົາຄວນເລືອກໄປເສັ້ນທາງໃດກ່ອນ ໃນກໍລະນີມີ ເສັ້ນທາງຫຼາຍກວ່າໜຶ່ງເສັ້ນທາງຕ້ອງເລືອກ.

ຂະບວນການຄົ້ນຫາແບບຮິວຣິສະຕິກ ໂດຍປົກກະຕິແລ້ວຈະຕ້ອງອາໄສຮິວຣິສະຕິກຟັງຊັນ ເຮັດ ໃຫ້ການແກ້ປັນຫາໜຶ່ງໆ ຈະດີ ຫຼື ບໍ່ ກໍຂຶ້ນຢູ່ກັບຮິວຣິສະຕິກຟັງຊັນດັ່ງນັ້ນການຄົ້ນຫາແບບນີ້ຈຶ່ງບໍ່ມີຫຍັງ ຮັບຮອງວ່າຈະໄດ້ສິ່ງທີ່ບໍ່ດີອອກມາດ້ວຍເຫດນີ້ເອງ ເຮົາຈຶ່ງເອີ້ນການຄົ້ນຫາແບບຮິວຣິສະຕິກນີ້ວ່າ Weak Methods ຫຼື ເວົ້າໄດ້ອີກແບບຄື Weak Methods ເປັນຂະບວນການຄວບຄຸມໂດຍທົ່ວໄປ (General-Purpose Control Strategies) ເຊິ່ງການຄົ້ນຫາແບບຮິວຣິສະຕິກທີ່ສຳຄັນມີດັ່ງຕໍ່ໄປນີ້ດັ່ງຕໍ່ໄປນີ້.

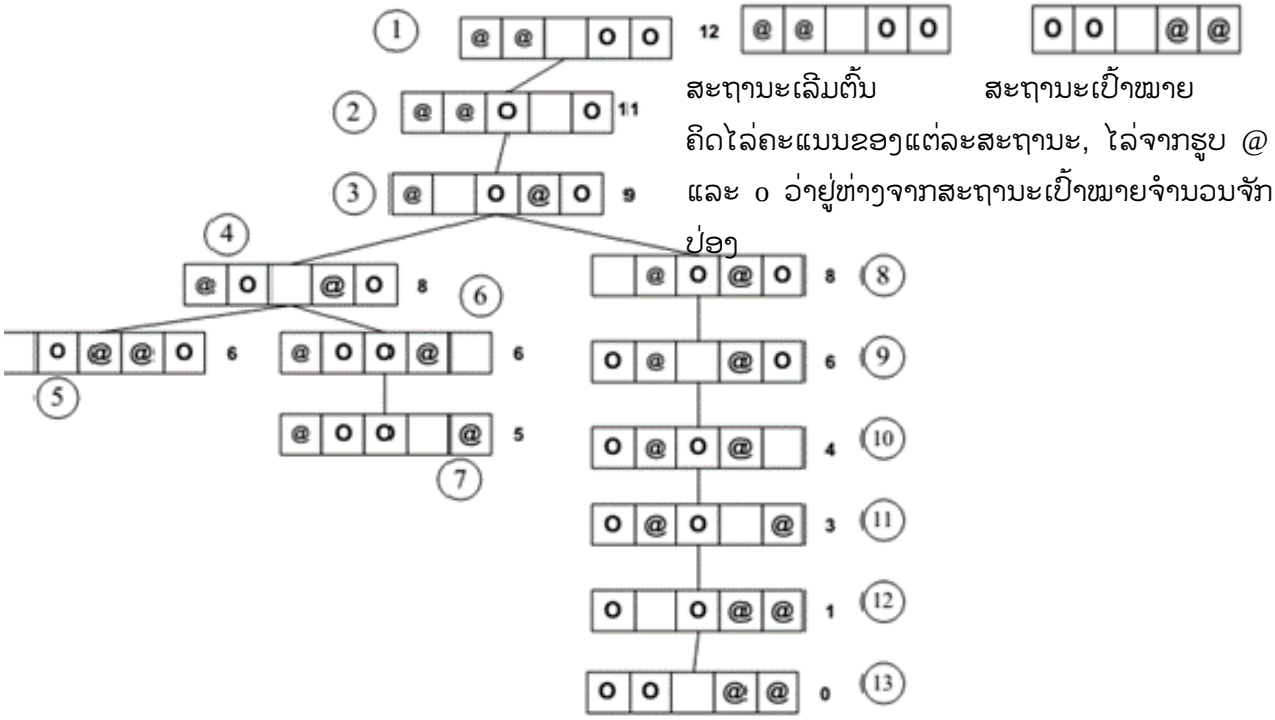
2.3.1. ໄຕ່ຂຶ້ນພູ (Hill Climbing)

ການຄົ້ນຫາແບບໄຕ່ຂຶ້ນພູ (Hill Climbing) ເປັນວິທີການຄົ້ນຫາຂໍ້ມູນທີ່ມີລັກສະນະຄືກັບການໄຕ່ຂຶ້ນພູ ສິ່ງທີ່ນັກໄຕ່ຂຶ້ນພູຈະເດີນທາງໄປຮອດຍອດພູເຂົາ ນັກປີນເຂົາຈະຕ້ອງເບິ່ງກ່ອນວ່າຍອດເຂົາຢູ່ທີ່ ໃດແລ້ວນັກປີນເຂົາຈະຕ້ອງພະຍາຍາມໄປຈຸດນັ້ນໃຫ້ໄດ້ ລອງນຶກພາບຂອງການໄຕ່ຂຶ້ນພູຫ໊ອໂລ້ນທີ່ເບິ່ງ ເຫັນແຕ່ຍອດ ແລະ ນັກປີນເຂົາກຳລັງໄຕ່ຂຶ້ນພູຢູ່ທາງລຸ່ມທີ່ມີເສັ້ນທາງເຕັມໄປໝົດ ເພື່ອທີ່ຈະເດີນທາງໄປ ຮອດຍອດພູເຂົາໂດຍໄວທີ່ສຸດນັກປີນເຂົາຈະເບິ່ງໄປທີ່ຍອດເຂົາແລ້ວສັງເກດວ່າທິດທາງໃດທີ່ເມື່ອປີນແລ້ວ ຈະເຂົ້າໃກ້ຍອດເຂົາ ແລະຫຼີກລ້ຽງທິດທາງທີ່ໄປແລ້ວຈະເຮັດໃຫ້ຕົນເອງຫ່າງຈາກຍອດເຂົາ ນັກປີນເຂົາຈະ ຕ້ອງເຮັດແບບນີ້ໄປເລື້ອຍໆ ຈົນກວ່າຮອດຍອດເຂົາ.

ຄວາມຈິງແລ້ວການຄົ້ນຫາວິທີນີ້ເປັນວິທີທີ່ດັດແປງມາຈາກ Generate-and-Test ຕ່າງກັນທີ່ວິທີ ການຂອງ Generate-and-Test ໃຫ້ຄຳຕອບຂອງຟັງຊັນກວດສອບ (Test Function) ອອກມາເປັນ yes ແລະ no ແຕ່ຂອງ Hill Climbing ຈະໃຫ້ຄ່າອອກມາເປັນວ່າບ່ອນໃດໃກ້ທີ່ສຸດ, ເຊິ່ງປະກອບດ້ວຍຂັ້ນ ຕອນຕ່າງໆ ດັ່ງຕໍ່ໄປນີ້:

ກ. Simple Hill Climbing

1) ກວດສອບສະຖານະເລີ່ມຕົ້ນ ຖ້າສະຖານະເລີ່ມຕົ້ນນີ້ຄືສະຖານະເປົ້າໝາຍ ກໍໃຫ້ສະແດງຄຳຕອບ ອອກມາ ແລະຍົກເລີກການເຮັດວຽກ, ແຕ່ຖ້າສະຖານະເລີ່ມຕົ້ນບໍ່ໄດ້ເປັນສະຖານະເປົ້າໝາຍ ກໍ ປ່ຽນສະຖານະນີ້ເປັນສະຖານະປັດຈຸບັນ (Current State) ແລະ ໄປຕໍ່ຂໍ້ 2
2) ໃຫ້ເຮັດຕາມຂະບວນການຫາລຸ່ມນີ້ຈົນກວ່າຈະພົບຄຳຕອບ ຫຼື ຈົນກວ່າບໍ່ມີຕົວດຳເນີນການໃດໆທີ່ ຈະໃຊ້ກັບສະຖານະປັດຈຸບັນ ເພື່ອສ້າງສະຖານະໃໝ່
    2.1) ເລືອກຕົວດຳເນີນການທີ່ຍັງບໍ່ໄດ້ນຳມາໃຊ້ກັບສະຖານະປັດຈຸບັນ ມາໃຊ້ກັບສະຖານະ ປັດຈຸບັນເພື່ອສ້າງສະຖານະໃໝ່ຂຶ້ນມາສະຖານະໜຶ່ງ
    2.2) ກວດສອບສະຖານະໃໝ່ດັ່ງນີ້:
        - ຖ້າເປັນຄຳຕອບໃຫ້ເລີກການເຮັດວຽກ
        - ຖ້າບໍ່ແມ່ນ ແຕ່ວ່າສະຖານະທີ່ໄດ້ໃໝ່ດີກວ່າສະຖານະປັດຈຸບັນ ໃຫ້ສະຖານະໃໝ່ນີ້ເປັນ ສະຖານະປັດຈຸບັນ
        - ຖ້າບໍ່ແມ່ນ ແລະ ສະຖານະທີ່ໄດ້ບໍ່ດີເທົ່າສະຖານະປັດຈຸບັນ ໃຫ້ກັບເຮັດຊ້ຳທີ່ຂໍ້ 2 ໃໝ່



ຮູບທີ 2.14 ການແກ້ປັນຫາດ້ວຍ Hill Climbing

ຂ. ໄຕ່ຂຶ້ນຕາມທາງຊັນທີ່ສຸດ (Steepest-Ascent Hill Climbing)

1) ກວດສອບສະຖານະເລີ່ມຕົ້ນ ຖ້າສະຖານະເລີ່ມຕົ້ນນີ້ຄືສະຖານະເປົ້າໝາຍ ກໍໃຫ້ສະແດງຄຳຕອບອອກມາ ແລະຍົກເລີກການເຮັດວຽກ, ແຕ່ຖ້າສະຖານະເລີ່ມຕົ້ນບໍ່ໄດ້ເປັນສະຖານະເປົ້າໝາຍ ກໍປ່ຽນສະຖານະນີ້ເປັນສະຖານະປັດຈຸບັນ (current state) ແລະເຮັດຕໍ່ທີ່ຂໍ້ 2

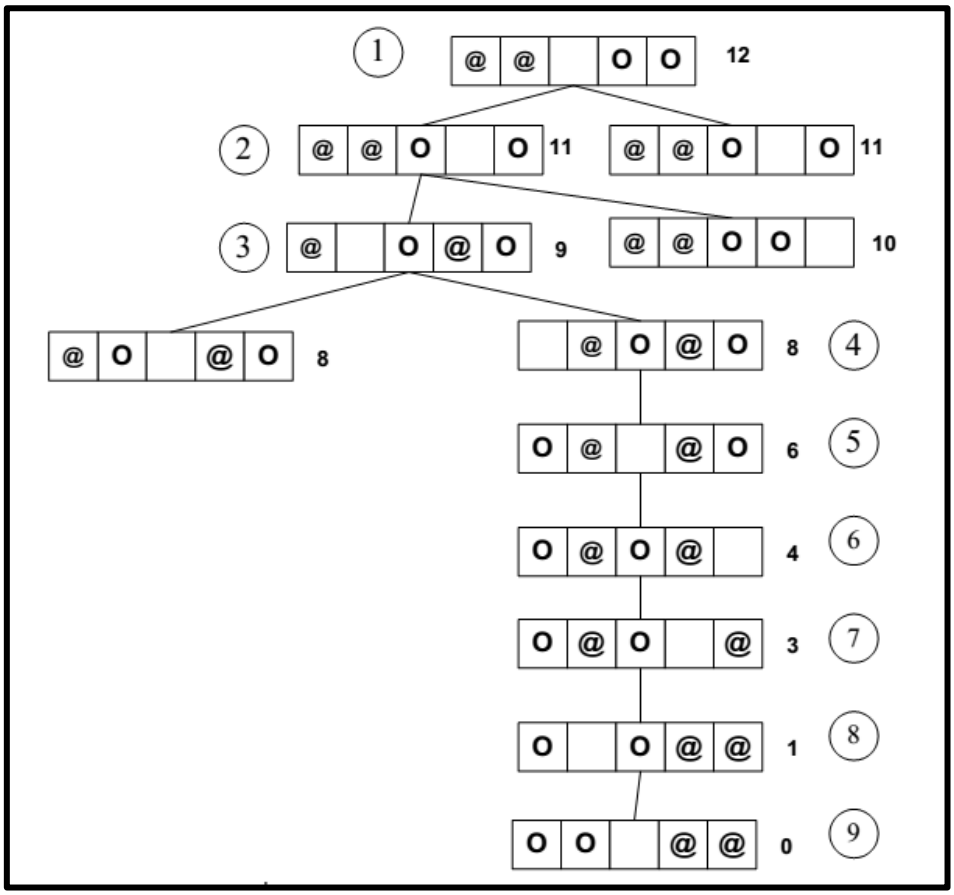
2) ໃຫ້ທຳຕາມຂະບວນການຂ້າງລຸ່ມນີ້ຈົນກວ່າຈະພົບຄຳຕອບ ຫຼື ຈົນກວ່າຄາດວ່າບໍ່ສາມາດຫາຄຳຕອບໄດ້

2.1) ໃຫ້ SUCC ເປັນສະຖານະທີ່ບໍ່ວ່າສະຖານະລູກໃດ ๆ ທີ່ເກີດໃໝ່ຈະຕ້ອງດີກວ່າສະຖານະນີ້

2.2) ສຳລັບຕົວດຳເນີນການທີ່ຈະໃຊ້ກັບສະຖານະປັດຈຸບັນແຕ່ລະຕົວ ໃຫ້ດຳເນີນການດັ່ງນີ້

- ສ້າງສະຖານະໃໝ່ຈາກສະຖານະປັດຈຸບັນ
- ກວດສອບສະຖານະໃໝ່ ຖ້າເປັນຄຳຕອບ ໃຫ້ເລີກການເຮັດວຽກ, ຖ້າບໍ່ແມ່ນໃຫ້ປຽບທຽບສະຖານະໃໝ່ກັບ SUCC ຖ້າສະຖານະທີ່ໄດ້ໃໝ່ດີກວ່າສະຖານະປັດຈຸບັນ ໃຫ້ສະຖານະໃໝ່ນີ້ເປັນ SUCC ຖ້າບໍ່ດີກວ່າ ບໍ່ຕ້ອງເຮັດຫຍັງໝົດ

2.3) ຖ້າ SUCC ດີກວ່າສະຖານະປັດຈຸບັນ ໃຫ້ສະຖານະປັດຈຸບັນຄື SUCC



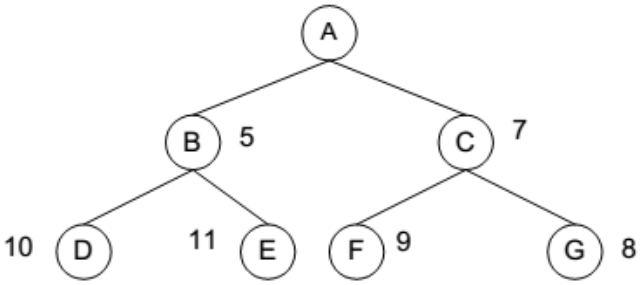
ຮູບທີ 2.15 ສະແດງການແກ້ປັນຫາແບບໄຕ່ຂຶ້ນຕາມທາງຊັນທີ່ສຸດ

ຄວາມແຕກຕ່າງລະຫວ່າງ Hill climbing ທຳມະດາ ກັບແບບ Steepest-Ascent ຄືແບບທຳມະດາຈະເລືອກສະຖານະປັດຈຸບັນຈາກໂນດລູກທີ່ດີກວ່າສະຖານະປັດຈຸບັນຕົວທຳອິດທີ່ພົບໂດຍບໍ່ສົນໃຈລູກທີ່ເຫຼືອ ແລ້ວສ້າງໂນດຂອງລຸ້ນຖັດໄປເພື່ອປຽບທຽບຕໍ່ໄປອີກ ແຕ່ສຳລັບ steepest ascent ນັ້ນຈະເລືອກສະຖານະປັດຈຸບັນຈາກຕົວທີ່ດີທີ່ສຸດຂອງລູກໝົດຊຸດ

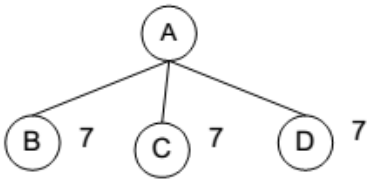
ການແກ້ປັນຫາດ້ວຍ Hill climbing ນັ້ນ ເຖິງວ່າຈະມີຂໍ້ດີຫຼາຍຢ່າງຄ້າຍຄື Generate and Test ແລ້ວ ໃນໂຕຂອງມັນເອງຍັງມີປັນຫາທີ່ອາດຈະເກີດຂຶ້ນໃນການໃຊ້ວິທີການນີ້ຄື:

- Local Maximum: ເປັນສ່ວນທີ່ບອກສະຖານະທີ່ດີທີ່ສຸດເມື່ອທຽບກັບສະຖານະຂ້າງຄຽງເທົ່ານັ້ນ ແຕ່ຖ້າທຽບກັບສະຖານະອື່ນໆ ທີ່ຢູ່ຫ່າງອອກໄປ ຫຼື ການກວດສອບຂັ້ນຕໍ່ໄປແລ້ວເປັນໄປໄດ້ວ່າຈະໄດ້ຜົນອອກມາດີທີ່ສຸດ ຈາກຮູບ (a) ເຮົາຈະເຫັນວ່າທີ່ໂນດໃນລະດັບທີ 1 ໂນດ B ແລະ C ຄ່າຂອງ B ນ້ອຍກວ່າ C ແຕ່ວ່າລູກຂອງ B ກັບມີຄ່າຫຼາຍກວ່າລູກຂອງ C ດັ່ງນັ້ນໃນການເລືອກໂນດທີ່ທີ່ສຸດໃນລະດັບທີ 1 ຄື C ຈະມີຜົນໃຫ້ໂນດ E ເຊິ່ງຄວນຈະເປັນໂນດທີ່ດີທີ່ສຸດ ບໍ່ໄດ້ຮັບການພິຈາລະນາ ເພາະໃນລະດັບເທິງ ຫຼື ລະດັບທີ 1 ໂນດ Bບໍ່ໄດ້ຮັບການເລືອກ
- ພູພຽງ (Plateau): ໃນກໍລະນີປັນຫາຢູ່ເທິງລະດັບດຽວກັນ ແລະ ຜົນຂອງການຫາຄ່າຈາກຮິວຣິສະຕິກ

ຟັງຊັນທີ່ໄດ້ເທົ່າກັນໝົດ ຈະບໍ່ສາມາດຕັດສິນໃຈໄດ້ວ່າຈະເລືອກເສັ້ນທາງເສັ້ນໃດ ຈາກຮູບ (b) ຈະເຫັນວ່າໂນດລູກທັງ 3 ຂອງ A ມີຄ່າເທົ່າກັນໝົດເຮັດໃຫ້ເຮົາບໍ່ສາມາດເລືອກໂນດທີ່ດີທີ່ສຸດໄດ້



(a) ການຄົ້ນຫາທີ່ເກີດປັນຫາແບບ Local Maximum

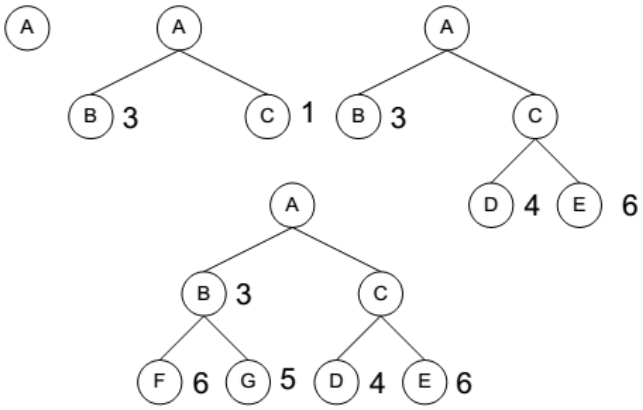


(b) ການຄົ້ນຫາທີ່ເກີດປັນຫາແບບ Plateua

- ສັນເຂົາ (Ridge): ເປັນກໍລະນີການຄົ້ນຫາໄດ້ໄປໃນທາງທີ່ດີທີ່ສຸດຕະຫຼອດເວລາ ເມື່ອປຽບທຽບກັບໂນດຂ້າງຄຽງ ແຕ່ເມື່ອທຽບກັບໂນດທີ່ດີທີ່ສຸດໃນລະດັບທີ່ແລ້ວຈະບໍ່ດີ ການຄົ້ນຫາໃນລັກສະນະນີ້ຈະມີລັກສະນະແບບດຽວກັບການໄປເທິງສັນເຂົານ້ອຍໆ ທີ່ເບິ່ງຄືວ່າຈະພາໄປສູ່ຍອດເຂົາໄດ້.

2.3.2. ການຄົ້ນຫາແບບດີທີ່ສຸດກ່ອນ(Best-First Search)

ເປັນຂະບວນການຄົ້ນຫາຂໍ້ມູນທີ່ໄດ້ນຳເອົາຂໍ້ດີຂອງທັງການຄົ້ນຫາແບບເລິກກ່ອນ(Depth First Search) ແລະການຄົ້ນຫາແບບລວງກວ້າງກ່ອນ(Breadth First Search) ມາລວມກັນເປັນວິທີການດຽວ ໂດຍທີ່ແຕ່ລະຂັ້ນຂອງການຄົ້ນຫາໃນໂນດລູກນັ້ນ ການຄົ້ນຫາແບບດີທີ່ດີກ່ອນຈະເລືອກເອົາ ໂນດທີ່ດີທີ່ສຸດ (Most Promising) ແລະ ການທີ່ຈະຮູ້ວ່າໂນດໃດດີທີ່ສຸດນີ້ສາມາດເຮັດໄດ້ໂດຍອາໄສຮິວຣິສະຕິກຟັງຊັນ ເຊິ່ງຮິວຣິສະຕິກຟັງຊັນ ນີ້ຈະເຮັດໜ້າທີ່ຄືຕີວັດຜົນ ແລະ ໃຫ້ຜົນຂອງການວັດນີ້ອອກມາເປັນຄະແນນ ຮູບທີ 2.7 ເປັນຕົວຢ່າງຂອງການຄົ້ນຫາແບບດີທີ່ສຸດກ່ອນ ຂັ້ນຕອນນີ້ເລີ່ມຈາກຕອນ 1 ສ້າງໂນດຮາກ (Root Node) ໃນຂັ້ນຕອນ 2 ສ້າງໂນດລູກ B ແລະ C ແລ້ວກວດສອບໂນດ B ແລະ C ດ້ວຍຮິວຣິສະຕິກຟັງຊັນ ໄດ້ຜົນອອກມາເປັນຄະແນນຄື 3 ແລະ 1 ຕາມລຳດັບ ຈາກນັ້ນໃຫ້ເລືອກໂນດ C ເປັນໂນດຕໍ່ໄປທີ່ເຮົາສົນໃຈ ເພາະມີຄ່າໜ້ອຍກວ່າ (ໝາຍເຫດ ໃນການເລືອກນີ້ຈະເລືອກຄ່າຫຼາຍສຸດ ຫຼືໜ້ອຍສຸດກໍໄດ້ ຂຶ້ນຢູ່ກັບລັກສະນະຂອງປັນຫາ) ແລ້ວສ້າງໂນດ ລູກໃຫ້ກັບໂນດ C ໃນຂັ້ນຕອນ 3 ໄດ້ໂນດ D ແລະ E ແລ້ວກວດສອບຄະແນນໄດ້ 4 ແລະ 6 ຕາມລຳດັບ ຈາກນັ້ນສ້າງການ ປຽບທຽບຄ່າຂອງໂນດທ້າຍສຸດ ຫຼື Terminal Node ທຸກໂນດ ວ່າໂນດ ໃດມີຄ່າດີທີ່ສຸດ ໃນທີ່ນີ້ຈະຕ້ອງເລືອກໂນດ B ເພາະມີຄະແນນພຽງ 3 (ເລືອກຄະແນນຕ່ຳສຸດ) ແລ້ວສ້າງໂນດ ລູກຕາມຂັ້ນຕອນ 4 ໄດ້ F ແລະ G ແລ້ວກວດສອບຄະແນນໄດ້ 6 ແລະ 5 ຄະແນນຕາມລຳດັບ ເຮັດແບບນີ້ເລື້ອຍໆຈົນພົບຄຳຕອບ ຫຼື ຈົນບໍ່ສາມາດ ສ້າງໂນດຕໍ່ໄປໄດ້ອີກ.



ຮູບທີ 2.16 ການສະແດງຂັ້ນຕອນຂອງການຄົ້ນຫາແບບດີທີ່ສຸດກ່ອນ

- **ອານກໍຣິດທຶມ: ການຄົ້ນຫາແບບດີທີ່ສຸດກ່ອນ**

1) ເລີ່ມດ້ວຍ OPEN ທີ່ມີພຽງໂນດເລີ່ມຕົ້ນ
2) ຈົນກວ່າຈະພົບເປົ້າໝາຍ ຫຼືວ່າບໍ່ມີໂນດເຫຼືອຢູ່ໃນ OPEN

- ເລືອກໂນດທີ່ດີທີ່ສຸດໃນ OPEN
- ສ້າງໂນດລູກໃຫ້ກັບໂນດທີ່ດີທີ່ສຸດນັ້ນ
- ສຳລັບໂນດລູກແຕ່ລະຕົວໃຫ້ທຳດັ່ງຕໍ່ໄປນີ້:
    - ຖ້າໂນດນັ້ນຍັງບໍ່ເຄີຍຖືກສ້າງມາກ່ອນໜ້ານັ້ນ ໃຫ້ກວດສອບຄ່າຂອງມັນໂດຍໃຊ້ຮິວຣິສະຕິກຟັງຊັນແລ້ວເພີ່ມເຂົ້າໄປໃນ OPEN ແລ້ວບັນທຶກວ່າເປັນໂນດແມ່
    - ຖ້າໂນດນັ້ນຖືກສ້າງມາກ່ອນໜ້ານີ້ແລ້ວ ໃຫ້ປ່ຽນໂນດແມ່ຂອງມັນ ຖ້າເສັ້ນທາງໃໝ່ທີ່ໄດ້ດີກວ່າໂນດແມ່ຕົວເດີມ ໃນກໍລະນີນີ້ ໃຫ້ປັບປ່ຽນຄ່າຕາມເສັ້ນ ທາງທີ່ອາດຈະເກີດຂຶ້ນ

2.3.4. ອານກໍຣິດທຶມ A*

ການຄົ້ນຫາແບບ A* ເປັນອີກແບບຂອງການຄົ້ນຫາແບບດີທີ່ສຸດກ່ອນ ວິທີການເລືອກໂນດທີ່ຈະໃຊ້ໃນການດຳເນີນການຕໍ່ຈະພິຈາລະນາຈາກໂນດທີ່ດີທີ່ສຸດ ແຕ່ໃນກໍລະນີຂອງ A* ນີ້ຈະມີລັກສະນະພິເສດກວ່າຄື ໃນສ່ວນຂອງຮິວຣິສະຕິກຟັງຊັນໃນກໍລະນີຂອງການຄົ້ນຫາແບບດີທີ່ສຸດກ່ອນນັ້ນຄ່າທີ່ໄດ້ຈາກຮິວຣິສະຕິກຟັງຊັນຈະເປັນຄ່າທີ່ໄດ້ຈາກ ໂນດປັດຈຸບັນ ແຕ່ໃນກໍລະນີຂອງ A* ຄ່າຂອງຮິວຣິສະຕິກຟັງຊັນ ຈະວັດຈາກຄ່າ 2 ຄ່າຄື ຄ່າທີ່ແທກຈາກໂນດປັດຈຸບັນໄປຍັງໂນດຮາກ ແລະ ຈາກໂນດປັດຈຸບັນໄປຍັງໂນດເປົ້າໝາຍ ຖ້າເຮົາໃຫ້ຕົວປ່ຽນ f ແທນຄ່າຂອງຮິວຣິສະຕິກ ຟັງຊັນ g ເປັນຟັງຊັນທີ່ໃຊ້ວັດຄ່າ Cost ຈາກສະຖານະເລີ່ມຕົ້ນຈົນຮອດສະຖານະປັດຈຸບັນ h' ເປັນຟັງຊັນທີ່ໃຊ້ວັດຄ່າ Cost ຈາກສະຖານະປັດຈຸບັນຮອດສະຖານະເປົ້າໝາຍດັ່ງນັ້ນ

$$f = g + h'$$

- **ອານກໍຣິດທຶມແບບ A*(A* algorithm)**

ໃນໂນດໜຶ່ງໆຂອງການຄົ້ນຫາແບບ A*ຈະມີອົງປະກອບດັ່ງຕໍ່ໄປນີ້

- ຕົວອະທິບາຍ(Description)ທີ່ສະແດງວ່າໂນດນັ້ນຄືຫຍັງ
- ອິນດິເຄເຕີ (Indicator)ທີ່ສະແດງຄວາມສຳຄັນຂອງໂນດ
- ຕົວຊີ້(Pointer)ທີ່ຊີ້ໄປຍັງ Parent ຂອງມັນ
- ລິດ (List)ຂອງໂນດທີ່ມັນສ້າງຂຶ້ນ

ຕົວ Parent-Link ຈະເປັນສ່ວນທີ່ເຮັດໃຫ້ເກີດການ Recover ຂອງເສັ້ນທາງໄປຍັງເປົ້າໝາຍເມື່ອພົບລິດຂອງໂນດລູກທີ່ສະແດງເສັ້ນທາງທີ່ດີທີ່ສຸດ

ສຳລັບໂນດລູກໜຶ່ງຈະຖືກຈັດອອກເປັນ 2 ປະເພດຄື:

- OPEN ຄືຕົວປ່ຽນເຮັດໜ້າທີ່ເປັນຄິວທີ່ໃຊ້ສຳລັບເກັບໂນດທີ່ສ້າງຂຶ້ນມາແລະຖືກກວດສອບດ້ວຍຮິວຣິສະຕິກຟັງຊັນແລ້ວ ແຕ່ຍັງບໍ່ໄດ້ມີການເຮັດວຽກອື່ນຕໍ່ເຊັ່ນ: ການສ້າງໂນດລູກ, ລາຍການນີ້ຈະເປັນຕາຕະລາງຂອງ Priority Queue ດັ່ງນັ້ນໃນການຈັດການກ່ຽວກັບເລື່ອງນີ້ຈະສາມາດໃຊ້ເທກນິກໃນການແກ້ປັນຫາຂອງ QUEUE ມາໃຊ້ໄດ້
- CLOSE ຄືຕົວປ່ຽນເຮັດໜ້າທີ່ສຳລັບເກັບໂນດທີ່ຜ່ານການເຮັດວຽກແລ້ວ

ສຳລັບຮິວຣິສະຕິກຟັງຊັນຂອງຂະບວນການນີ້ກໍຄືການກວດສອບຄ່າຂອງຄວາມສຳຄັນເຊິ່ງຈະໃຊ້ຕົວຍ່ອງ ເປັນ f' ແລະ f' ນີ້ຍັງຖືກແບ່ງອອກເປັນ 2 ສ່ວນຄື:

- g =ຟັງຊັນທີ່ໃຊ້ວັດຄ່າ costຈາກສະຖານະເລີ່ມຕົ້ນຈົນຮອດສະຖານະປັດຈຸບັນ
- h'=ຟັງຊັນທີ່ໃຊ້ວັດຄ່າ cost ຈາກສະຖານະປັດຈຸບັນຮອດສະຖານະເປົ້າໝາຍ

ດັ່ງນັ້ນ f=g+h' ທີ່ດີທີ່ສຸດເທົ່ານັ້ນ ແລະ g ຈະມີຄ່າເປັນລົບບໍ່ໄດ້ ເພາະການລົບລູບໃນກຣາບ ຈະສ້າງໃຫ້ໄດ້ຄ່າທີ່ດີຂຶ້ນ ເຊິ່ງຜິດກັບຄວາມເປັນຈິງ.

- **ການເຮັດວຽກຂອງອານກໍຣິດທຶມ**

1) ເລີ່ມໂດຍໃຫ້ OPEN ມີພຽງໂນດຣາກ (Initial Node) ເທົ່ານັ້ນ, ໃນຊ່ວງທຳອິດທີ່ໂນດປັດຈຸບັນແມ່ນໂນດຣາກ, ດັ່ງນັ້ນຄ່າທີ່ໄດ້ຮັບຈາກໂນດຣາກຮອດໂນດປັດຈຸບັນກໍຄື 0 ນັ້ນຄື g = 0 ແລະ h' ເປັນຄ່າທີ່ກວດສອບໄດ້ ດັ່ງນັ້ນ f=h'+0 ແລ້ວໃຫ້ໂນດຣາກນີ້ໃສ່ໃນ CLOSED
2) ໃຫ້ເຮັດດັ່ງຕໍ່ໄປນີ້ຈົນກວ່າພົບເປົ້າໝາຍ:
   2.1) ຖ້າບໍ່ມີໂນດໃດໆ ທີ່ OPEN ຢູ່ເລີຍ ໃຫ້ລາຍງານວ່າ Fail
   2.2) ຖ້າມີໃຫ້ເລືອກໂນດໃນ OPEN ທີ່ມີຄ່າຂອງ f ຕ່ຳສຸດເປັນ BESTNODE ແລ້ວດຶງອອກຈາກລາຍການຂອງ OPEN ພ້ອມກັບ CLOSED ໂນດນັ້ນແລ້ວກວດສອບວ່າ BESTNODE ນີ້ເປັນເປົ້າໝາຍໄດ້ ຫຼືບໍ່ ຖ້າໄດ້ກໍຖືວ່າການຄົ້ນຫາສຳເລັດ
   2.3) ຖ້າບໍ່ໄດ້ໃຫ້ສ້າງໂນດລູກຂອງ BESTNODE ໃໝ່ໂດຍທີ່ຍັງບໍ່ຕ້ອງເຊື່ອມ Pointer ຈາກ BESTNODE ຊີ້ມາບ່ອນນີ້ສຳລັບໂນດລູກຂອງ BESTNODE ແຕ່ລະຕົວໃຫ້ເຮັດດັ່ງນີ້:
   - ໃຫ້ໂນດລູກຊີ້ກັບໄປທີ່ BESTNODE ການເຊື່ອມນີ້ຈະເປັນຕົວທີ່ສ້າງໃຫ້ສາມາດໄລ່ເສັ້ນທາງທີ່ເທື່ອໜຶ່ງເຄີຍເປັນຄຳຕອບໄດ້

- ໃຫ້ຄຳນວນຄ່າ g(SUCCESSOR) = g(BESTNODE) + Cost ຈາກ BESTNODE ຮອດ SUCCESSOR
- ໃຫ້ກວດວ່າຄ່າ Cost ຂອງ SUCCESSOR ນີ້ມີ Cost ເທົ່າກັບ OPEN ໂນດອື່ນ ຫຼື ບໍ່ຖ້າ ເທົ່າໃຫ້ເອີ້ນໂນດນີ້ວ່າ OLD ແລະ ໃຫ້ຕັດ SUCCESSOR ນີ້ອອກ ແລະ ໃຫ້ OLD ນີ້ຢູ່ ໃນລິດ ຂອງ BESTNODE ຂອງໂນດລູກແລ້ວຕັດສິນວ່າ Parent Link ຂອງ OLD ຈະ ຕ້ອງຣີເຊດໃຫ້ຊີ້ໄປທີ່ BESTNODE ຫຼືບໍ່ ລົ້ງນີ້ຈະຕ້ອງຣີເຊດຖ້າເສັ້ນທາງໃໝ່ທີ່ໄດ້ຄິດແລ້ວດີ ທີ່ສຸດ (ທັ້ງນີ້ເພາະ SUCCESSOR ແລະ OLD ກໍຄືໂນດດຽວກັນ) ດັ່ງນັ້ນເຮົາຈຶ່ງຕ້ອງຄິດ ວ່າເສັ້ນທາງຈາກ Parent ປັດຈຸບັນຮອດ SUCCESSOR ກັບເສັ້ນທາງຈາກ BESTNODE ຮອດ SUCCESSOR ອັນໃດດີກວ່າກັນ ໂດຍການປຽບທຽບຄ່າ g ຖ້າ OLD ດີກວ່າຫຼືດີເທົ່າ ກັນ ກໍບໍ່ຕ້ອງເຮັດຫຍັງ ຖ້າ SUCCESSOR ດີກວ່າໃຫ້ຣີເຊດ Parent ຂອງ OLD ໃຫ້ລົ້ງຊີ້ ໄປທີ່ BESTNODE ແລະ ບັນທຶກເສັ້ນທາງທີ່ດີທີ່ສຸດໃນ g(OLD) ແລະ Update f(OLD)
- ຖ້າ SUCCESSOR ບໍ່ໄດ້ເປັນ OPEN ໃຫ້ເບິ່ງວ່າເປັນ CLOSE ຫຼືບໍ່ ຖ້າເປັນໃຫ້ເອີ້ນວ່າ OLD ແລ້ວລວມ OLD ນີ້ໃຫ້ຢູ່ໃນລາຍການຂອງ BESTNODE's Successor ກວດສອບ ວ່າເສັ້ນທາງໃດດີທີ່ສຸດ (ລະຫວ່າງ ໃໝ່ ແລະ ເກົ່າ) ດ້ວຍວິທີດຽວກັບຫົວຂໍ້ຂ້າງເທິງ ແລະ ເຊດ Parent ລົ້ງຄ່າ g ແລະຄ່າ f' ທີ່ເໝາະສົມ, ໃນກໍລະນີທີ່ OLD ເປັນເສັ້ນທາງທີ່ດີກວ່າ ເຮົາຕ້ອງເຄື່ອນ (Propagate) ທຸກຢ່າງໄປທີ່ໂນດລູກຂອງ OLD ໃຫ້ OLD ຊີ້ໄປ ທີ່SUCCESSOR ຂອງມັນ ແລະ ໂນດລູກແຕ່ລະຕົວຈະຕ້ອງຊີ້ໄປຍັງໂນດລູກຂອງມັນເອງ ອີກ ຈົນກວ່າຮອດໂນດສຸດທ້າຍ (Terminal Node) ຂອງແຕ່ລະກິ່ງບໍ່ວ່າໂນດນັ້ນຈະ OPEN ຫຼື ວ່າບໍ່ມີໂນດລູກໃນ ການໃສ່ (Propagate Cost) ໃໝ່ໃຫ້ກັບໂນດຕ່າງໆຂອງໂນດລູກນີ້ຈະ ເຮັດແບບ Depth-First Traverse ເທິງ Tree ເລີ່ມຈາກ OLD ປ່ຽນຄ່າ g ຂອງແຕ່ລະໂນດ (ແລະ ຄ່າ f') ໃຫ້ຢຸດການ Traverse ໃນທຸກກິ່ງທີ່ບໍ່ມີໂນດ ຫຼື ໂນດທີ່ມີຄ່າເສັ້ນທາງທີ່ເທົ່າ ກັບ ຫຼື ດີກວ່າທັງໝົດທີ່ໄດ້ຫາມາໃນ Parent Link ຂອງແຕ່ລະໂນດທີ່ຊີ້ກັບໄປຍັງ Parent ທີ່ດີທີ່ສຸດເທົ່າທີ່ຮູ້ ໃນຂະນະທີ່ເຄື່ອນລົງໄປທີ່ໂນດໃຫ້ກວດວ່າ Parent ຊີ້ໄປ ທີ່ໂນດທີ່ຫາກໍ່ ຜ່ານມາ ຫຼືບໍ່ ຖ້າແມ່ນໃຫ້ເຄື່ອນຕໍ່ ຖ້າບໍ່ແມ່ນສະແດງວ່າຄ່າ g ຂອງມັນມີຜົນຕໍ່ເສັ້ນທາງທີ່ດີທີ່ ສຸດທີ່ຫາກໍ່ຜ່ານມາ ດັ່ງນັ້ນການເຄື່ອນຈະຢຸດບ່ອນນີ້ ແຕ່ວ່າເປັນໄປໄດ້ທີ່ຄ່າໃໝ່ຂອງ g ຈະຖືກ ເຄື່ອນລົງເສັ້ນທາງ ທີ່ເຮົາກຳລັງທຳຢູ່ນີ້ ອາດຈະດີກວ່າເສັ້ນທາງທີ່ຜ່ານ Current Parent ໃຫ້ ປຽບທຽບທັງສອງອັນ ຖ້າຫາກວ່າທີ່ Current Parent ດີກວ່າໃຫ້ຢຸດເຄື່ອນ ຖ້າເສັ້ນທາງທີ່ເຮົາ ກຳລັງທຳຢູ່ດີກວ່າໃຫ້ຣີເຊດ Parent ແລະ ເຄື່ອນທີ່ຕໍ່ໄປ.
- ຖ້າ SUCCESSOR ບໍ່ໄດ້ເປັນທັງ OPEN ແລະ CLOSE ໃຫ້ OPEN ແລະລວມມັນເຂົ້າ ໃນລິດຂອງໂນດລູກຂອງ BESTNODE ແລະຄຳນວນ

$$f'(SUCCESSOR) = g(SUCCESSOR) + h'(SUCCESSOR)$$

ຈາກຕົວຢ່າງການຫາຄຳຕອບດ້ວຍການຄົ້ນຫາຂໍ້ມູນແບບ A*ຂອງປັນຫາ 8-Puzzle ໃນການຄຳນວນຄ່າຂອງ g ຄືການປຽບທຽບສະຖານະປັດຈຸບັນກັບສະຖານະເລີ່ມຕົ້ນແລະຄ່າ h' ຄືການປຽບທຽບສະຖານະປັດຈຸບັນກັບສະຖານະເປົ້າໝາຍເຊິ່ງໃນການຄຳນວນຄ່າ g ແລະ h'ຈະມີວິທີການດັ່ງນີ້

ສະຖານະເລີ່ມຕົ້ນ

2	8	3
1	6	4
7		5

ສະຖານະເປົ້າໝາຍ

1	2	3
4	5	6
7	8	9

ສະຖານະປັດຈຸບັນ

2		3
1	8	4
7	6	5

ສະຖານະເປົ້າໝາຍ

1	2	3
8		4
7	6	5

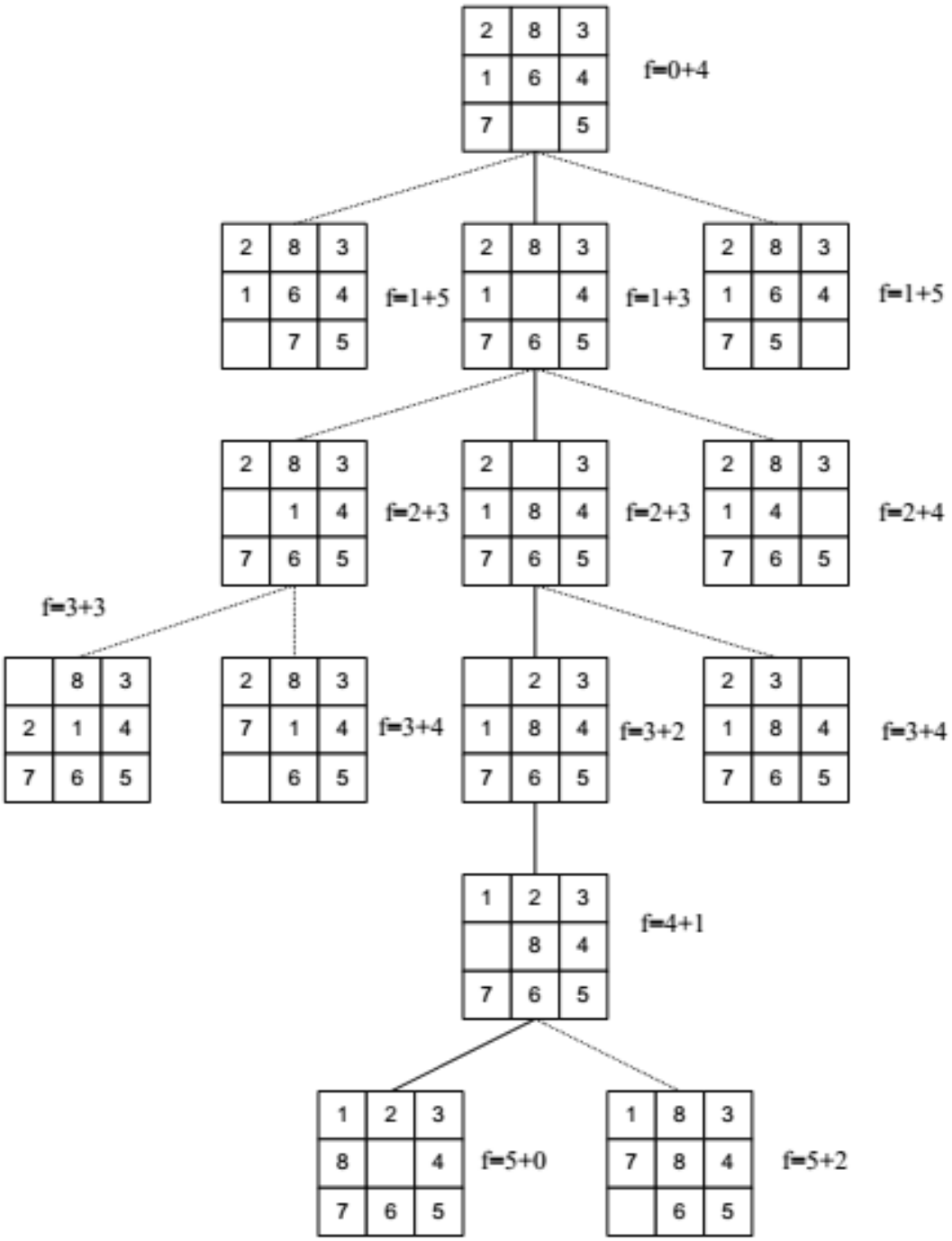
ການຫາຄ່າຂອງ g ເຮັດໄດ້ໂດຍການປຽບທຽບສະຖານະປັດຈຸບັນກັບສະຖານະເລີ່ມຕົ້ນ ຄື ການນັບວ່າແຕ່ລະປ່ອງຂອງຕາຕະລາງປັດຈຸບັນຫ່າງຈາກຈຸດເລີ່ມຕົ້ນໄປຈັກປ່ອງ ຄືປ່ອງທີ 1 ຕາຕະລາງທັງສອງບໍ່ຫ່າງຈາກກັນ ດັ່ງນັ້ນໄດ້ຄ່າເປັນ 0 ປ່ອງທີ 1 ປ່ອງທີ 2 ສະຖານະປັດຈຸບັນບໍ່ມີຄ່າບໍ່ຄິດ ແລະຄືກັນທີ່ປ່ອງທີ 3 ແລະ 4 ກໍມີຄ່າເປັນ 0 ເພາະ ທັງສອງຕາຕະລາງບໍ່ຕ່າງກັນ ສຳລັບຕາຕະລາງທີ 5 ຈະມີຄ່າເປັນ 1 ເພາະ 8 ຫ່າງຈາກທີ່ ທີ່ເຄີຍຢູ່ 1 ປ່ອງ ເຮັດແບບນີ້ເລື້ອຍໆ ເຮົາຈະໄດ້ຄ່າຕາມປ່ອງຕາຕະລາງຕ່າງໆ ດັ່ງນີ້

1=0, 2=0, 3=0, 4=0, 5=1, 6=0, 7=0, 8=1, 9=0 ລວມ g=2

ສຳລັບຄ່າຂອງ h' ຈະໄດ້ດັ່ງນີ້ 1

=1, 2=0, 3=0, 4=1, 5= 1, 6=0, 7=0, 8=0, 9=0 ລວມ h'=3

f = 2 + 3 = 5



ຮູບທີ 2.17 ແກ້ປັນຫາ 8 ປິດສະໜາ ດ້ວຍ A*

- **ຂໍ້ສັງເກດ**

1) ກ່ຽວກັບຕົວອານກໍຣິດທຶມ

- ໝາຍຄວາມວ່າເຮົາກຳລັງຄຳນຶງຫາເສັ້ນທາງທີ່ດີທີ່ສຸດ
- ໃນກໍລະນີທີ່ເຮົາຄຳນຶງສະເພາະວ່າຈະເຮັດແນວໃດທີ່ຈະໄດ້ຄຳຕອບເທົ່ານັ້ນເຮົາສາມາດກຳນົດໃຫ້ g=0 ໄດ້

- ຖ້າຕ້ອງການໃຫ້ໄດ້ເສັ້ນທາງທີ່ມີຂັ້ນຕອນໜ້ອຍທີ່ສຸດ ເຮົາສາມາດກຳນົດໃຫ້ຄ່າຂອງໂນດໄປຫາໂນດລູກມີຄ່າຄົງທີ່ (ເທົ່າກັບ1)

2) ກ່ຽວກັບຄ່າຂອງ h' ແລະ h (ຄ່າຈາກໂນດຮອດໂນດລູກ)

- ຖ້າ h'ເປັນ perfect estimator ຂອງ h, A* algorithm ກໍຈະກາຍເປັນການເຂົ້າຫາເປົ້າໝາຍໂດຍບໍ່ມີການຄົ້ນຫາ
- ເມື່ອ h'=ຄ່າທີ່ດີທີ່ສຸດກໍຄືການຫາກຳນົດໄລຍະທາງທີ່ໃກ້ທີ່ສຸດ
- ເມື່ອ h'= 0 ການຄົ້ນຫາຈະຖືກຄວບຄຸມໂດຍ g
- ຖ້າ g=0ອີກການຄົ້ນຫາຈະເປັນແບບ Random
- ຖ້າ gເປັນ 1 ການຄົ້ນຫາຈະເປັນ Breadth-First

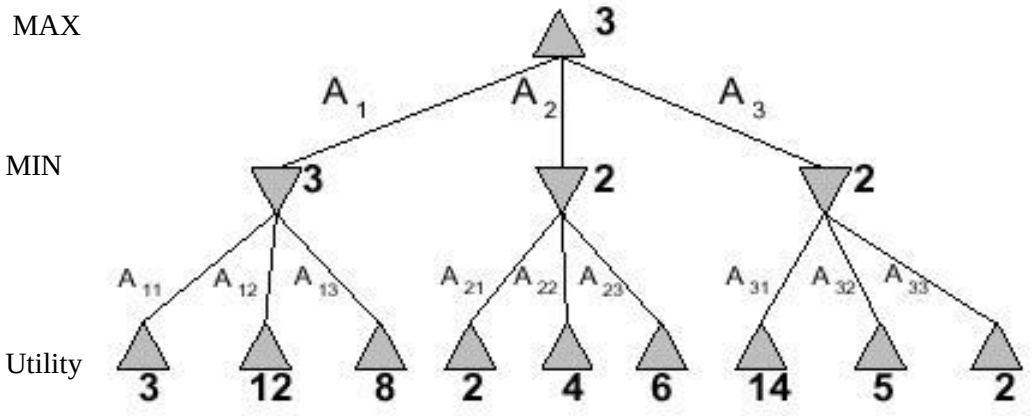
2.4. MINIMAX ALGORITHM

ມີການເຮັດວຽກແບບ Depth-first search ຄື ການສ້າງໂນດລົງໄປທາງເລິກກ່ອນ ຈົນຮອດລະດັບທີ່ຕ້ອງການ (ອາດຈະເປັນຈຸດສິ້ນສຸດຂອງເກມ ຫຼື ຈຸດລະດັບ N ໃດໜຶ່ງ) ໃຊ້ Utility Function ຫຼື Evaluation function ເພື່ອຫາຄ່າຄວາມດີຂອງໂນດນັ້ນ ສົ່ງຄ່າທີ່ໄດ້ກັບຂຶ້ນໄປໃນລະດັບທີ່ຢູ່ທາງເທິງເພື່ອຕັດສິນໃຈຕໍ່ໄປ ໃຊ້ເຕັກນິກການເອີ້ນໃຊ້ໂຕເອງ (Recursive) ໄດ້ ຄື Algorithm ຕໍ່ໄປນີ້

1. ສ້າງໂຄງຮ່າງຕົ້ນໄມ້ ທັງໝົດລົງໄປຫາ terminal node
2. ໃຊ້ utility function ເພື່ອຫາຄ່າຂອງໂນດນັ້ນ
3. ສົ່ງຄ່າຜົນຮັບກັບໄປຫາໂນດພໍ່ ຕາມກົດດັ່ງນີ້

   ຖ້າໂນດພໍ່ເປັນຜູ້ຫຼີ້ນ (MAX) ຈະຄືນຄ່າ Maximum ຂອງ ໂນດລູກ

   ຖ້າໂນດພໍ່ເປັນຜູ້ຫຼີ້ນຝ່າຍກົງຂ້າມ (MIN) ຈະຄືນຄ່າ Minimum ຂອງໂນດລູກ
4. MAX ຈະຖືກເລືອກ ໂນດທີ່ໃຫ້ຄ່າທາງໄປທີ່ສູງ (Maximum) ທີ່ສຸດ

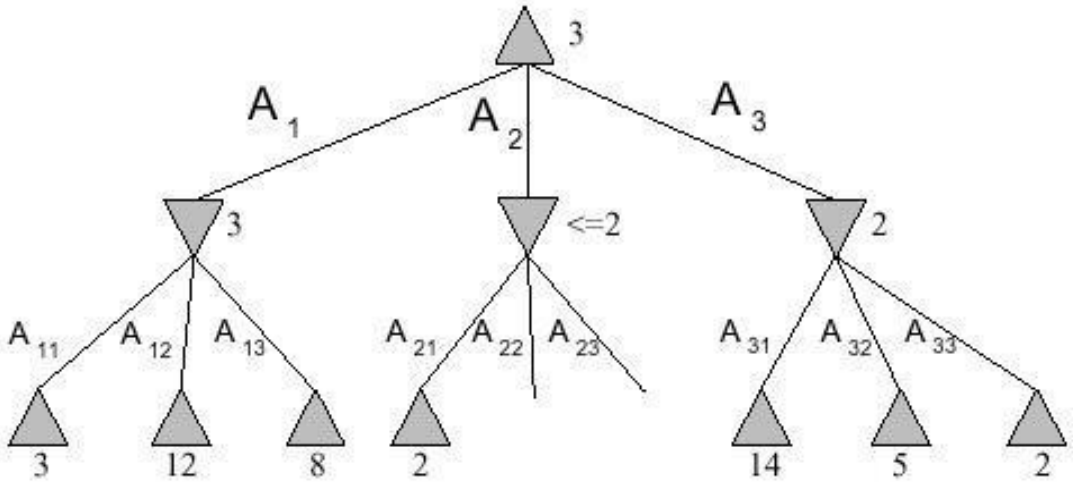


ຮູບທີ 2.18 ສະແດງເສັ້ນທາງ Minimax

Utility function ໃຫ້ຄ່າຄວາມດີທີ່ terminal node ຈາກນັ້ນ ທີ່ລະດັບ A1 ເຊິ່ງເປັນ MIN ຈະເລືອກຄ່າທີ່ນ້ອຍທີ່ສຸດ ສ່ວນໂນດ MAX ເທິ່ງສຸດຈະສົ່ງຄ່າໂນດລູກໃຫ້ຄ່າຫຼາຍທີ່ສຸດ

2.5. Alpha-Beta Cutoffs

ຫຼືເອີ້ນວ່າ Alpha-Beta Pruning ຄືການຕັດສາຂາທີ່ບໍ່ມີຜົນຕໍ່ການຕັດສິນໃຈອອກໄປ ເຮັດໃຫ້ຄຳນວນໄວຂຶ້ນ Algorithm ນີ້ປັບປຸງຈາກ Minimax Algorithm ໂດຍເພີ່ມການເຮັດວຽກສຳລັບເລືອກບໍ່ຄຳນວນບາງກິ່ງ ຄ່າ a ແລະ b ຖືກເພີ່ມລົງໄປແຕ່ລະໂນດທີ່ກຳລັງຄົ້ນຫາ ຄ່າ a ເກັບຄ່າທີ່ດີທີ່ສຸດຂອງ MAX ໂນດທີ່ເຄີຍເຫັນມາ ໂດຍທີ່ຄ່ານີ້ຈະບໍ່ມີທາງຫຼຸດລົງ (MAX ຕ້ອງການເລືອກຄ່າທີ່ຫຼາຍທີ່ສຸດ) ຄ່າ b ເກັບຄ່າທີ່ດີທີ່ສຸດຂອງ MIN ໂນດທີ່ເຄີຍເຫັນມາ ໂດຍທີ່ຄ່ານີ້ຈະບໍ່ມີທາງເພີ່ມຂຶ້ນ (MIN ຕ້ອງການເລືອກຄ່າທີ່ນ້ອຍທີ່ສຸດ)



ຮູບທີ 2.19 ສະແດງເສັ້ນທາງ Alpha-Bata Cutoff

- ກົດຂອງ Alpha-Beta Cutoffs

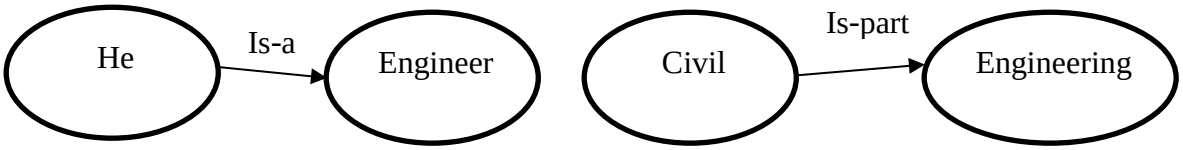
1. ການຄົ້ນຫາຈະຢຸດທີ່ໂນດໃດໜຶ່ງ ຂອງ MIN ເມື່ອຄ່າ ß <= A ຂອງໂນດ MAX ຊຶ່ງເປັນໂນດຂອງພໍ່

2. ການຄົ້ນຫາຈະຢຸດທີ່ໂນດຂອງ max ເມື່ອຄ່າ α >= B ຂອງໂນດ MIN ຊຶ່ງເປັນໂນດພໍ່

ບົດທີ 3 ການສະແດງຄວາມຮູ້ (knowledge Representation)

3.1 Semantic Network

Semantic Network ເປັນຮູບແບບການສະແດງຄວາມຮູ້ໂດຍໃຊ້ກາຣາບ ມາຊ່ວຍໃນການສືຄວາມໝ າຍ, Semantic network ຖືກພັດທະນາໂດຍຄິວລ້ຽນ (Quillian, 1968) ເພື່ອໃຊ້ແທນຄວາມໝາຍຂອງ ຄຳໃນພາສາອັງກິດ, ໂດຍການແທນຂ່າວສານ ແລະ ຂໍ້ມູນດ້ວຍໂນດທີ່ເຊື່ອມຕໍ່ກັນໂດຍອາກ (ລິ້ງ) ເຊິ່ງ ອາກແຕ່ລະອັນຈະລະບຸປ້າຍຊື່ທີ່ບອກເຖິງຄວາມສຳພັນຂອງໂນດແຕ່ລະໂນດທີ່ເຊື່ອມຕໍ່ກັນ.

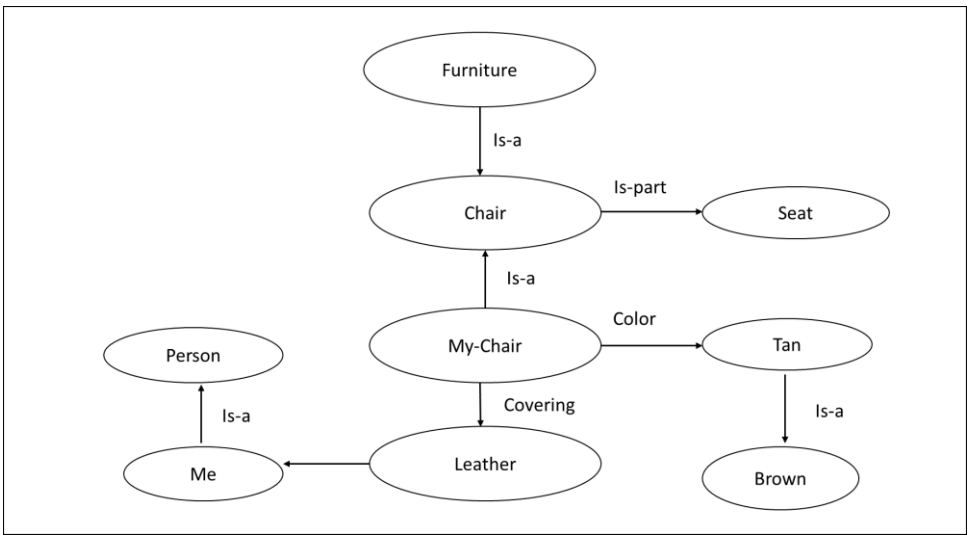


ຮູບທີ 3.1 ຄວາມຮູ້ຂອງປະໂຫຍກ He is an engineer ແລະ Civil is part of engineering

ຈາກຕົວຢ່າງໃນຮູບທີ 3.1 ສະແດງໃຫ້ເຫັນວິທີການສະແດງຄວາມຮູ້ແບບ Sematic Net ຂອງປະໂຫຍກ ‘ He is an engineer” ແລະ “Civil is part of engineering”ເຊິ່ງປະກອບດ້ວຍໂນດຂອງ He, Engineer, Civil Engineering ທີ່ຖືກເຊື່ອມຕໍ່ໂດຍ ອາກຂອງ is-a ແລະ is part ສະແດງຄວາມສຳພັນ ຂອງ He ແລະ engineer ກັບ Civil ແລະ engineering ຕາມລຳດັບ.

ສຳລັບນິຍາມຂອງໂນດ ແລະ ອາກ ຍັງບໍ່ມີການກຳນົດທີ່ແນ່ນອນ, ເຖິງຢ່າງໃດກໍ່ຕາມ ການສະແດງຄວາມຮູ້ແບບນີ້ ຈະມີຫຼັກການບາງຢ່າງທີ່ຮູ້ຈັກໂດຍທົ່ວໄປດັ່ງນີ້

- ໂນດ(Node) ໃຊ້ສະແດງເຖິງ ຊື່ຂອງວັດຖຸ (Object), ຄຸນສົມບັດ, ເຫດການ, ການເຮັດວຽກ
  - Object (ວັດຖຸ) ເຊັ່ນ Bird, Car, Basket-ball, Water, Glass, Cup, Table
  - Attribute ໃນນີ້ໝາຍເຖິງຄຸນສົມບັດຂອງວັດຖຸ ເຊັ່ນ Red, 20, good
  - Situation (ເຫດການ) ເປັນຄຳນາມທີ່ອະທິບາຍເຫດການ ເຊັ່ນ: Business-Crisis, Dead
  - Action (ການເຮັດວຽກ) ເຊັ່ນ Give, Go, Fly
- ອາກ (Arc) ສະແດງຄວາມໝາຍຂອງຄວາມສຳພັນລະຫວ່າງໂນດ. ບໍ່ມີກົດການຂຽນ Arc ຄືຈະ ຂຽນຄຳຫຍັງກໍ່ໄດ້ທີ່ສື່ເຖິງ Node ທີ່ສຳພັນກັນແຕ່ທີ່ນິຍົມໃຊ້ໄດ້ແກ່ Is-a, Has-part, Instance-of ຫຼື ອື່ນໆ



ຮູບທີ 3.2 ກຣາບສແດງ Sematic network

ວິທີການສະແດງຄວາມຮູ້ໃນຮູບທີ 6.2 ເປັນການສະແດງຄວາມຮູ້ໃນຮູບແບບຂອງກຣາບ ສຳລັບການສະແດງຄວາມຮູ້ຈິງໃນຄອມພິວເຕີນັ້ນ ເຮົາຈະໃຊ້ຮູບແບບຂອງຕັກກະສາດແບບໂປລັອກ (PROLOG) ແລະ ລິບ (LISP) ດັ່ງຕົວຢ່າງ ຕາຕະລາງ 3.1 ແລະ ຮູບທີ 3.3

Atom	Property List
Chair	((is-a Furniture))
My-Chair	((is-a Chair) (color Tan) (covering Leather) (owner Me))
Me	((is-a Person))
Tan	((is-a Brown))
Seat	((is-part Chair))

ຕາຕະລາງ 3.1 ການສະແດງຄວາມຮູ້ໂດຍໃຊ້ ລິບ (LISP)

Is-a (Chair, Furniture)

Is-a (Me, Person)

Covering (My-Chair, Leather)

Color (My-Chair, Tan)

ຮູບທີ 3.3 ການສະແດງຄວາມຮູ້ໂດຍໃຊ້ເພດິເຄດໂລຈິກ

3.1.1. Taxonomical Hierarchy

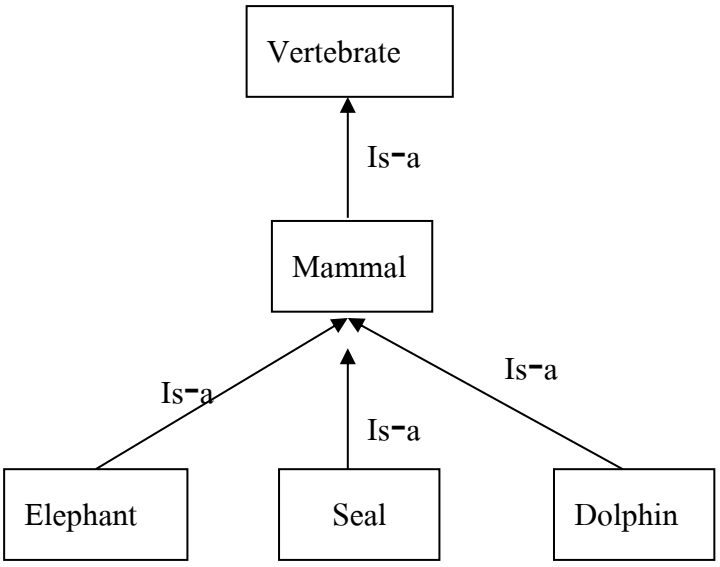
ເປັຍຂະບວນການໃນການຈັດລຳດັບຊັ້ນ(Hierarchy) ດັງທີສະແດງໃນຮູບ 3.4 ເຊິ່ງສະແດງການຈັດ class ຕ່າງຂອງ mammal ສ່ວນທີ່ມີຊັ້ນທີ່ສູງກວ່າເອີ້ນໄດ້ວ່າ Superclass ຄື Vertebrate ແລະ ຊັ້ນຕ່ຳກວ່າ ຫຼື Class ຍ່ອຍຄື Elephant, Seal ແລະ Dolphin ການເຊື່ອມຂອງໂນດທັງໝົດຈະເຫັນວ່າ ອາກມີຄວາມສຳພັນເປັນ is-a ເມື່ອເຮົາຂຽນໃນຮູບແບບຂອງເພດິເຄດໂລຈິກ ຈະໄດ້ດັ່ງລຸ່ມນີ້:

Is-a (Mammal, Vertebrate)

is-a (Elephant, Mammal)

Is-a (Seal, Mammal)

Is-a (Dolphin, Mammal)



ຮູບທີ 3.4 Sematic network

ສຳລັບ Taxonomical Hierarchy ສາມາດໃສ່ຄຸນສົມບັດຂອງໂນດເຂົ້າໄປນຳ ເຊັ່ນ:

do (Mammal, Giving-Milk)

do (Elephant, Clever)

do (Dolphin, Clever)

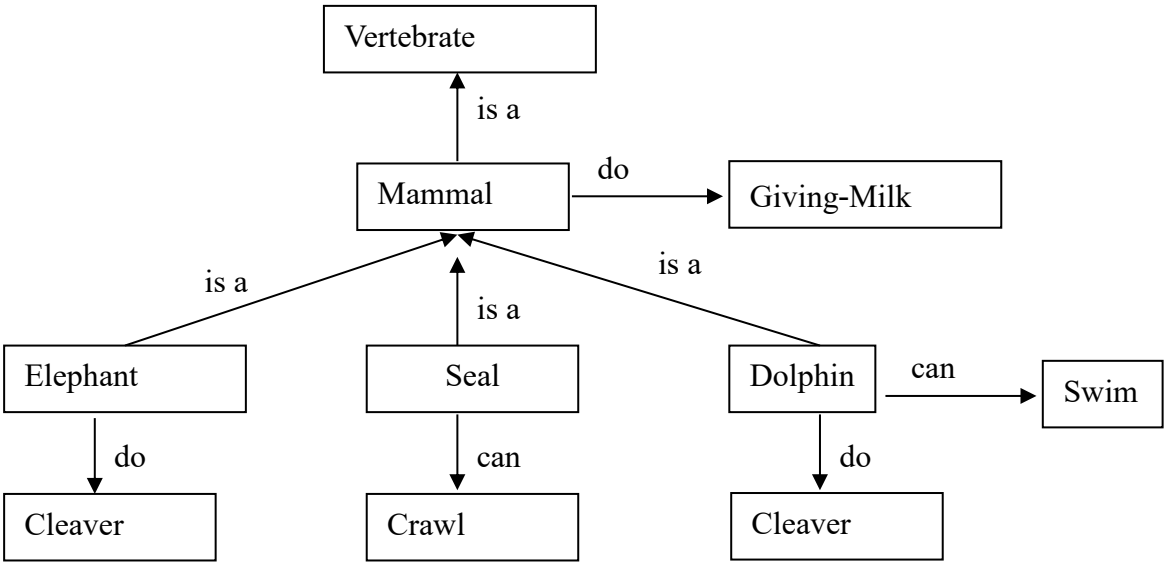
can (Elephant, Walk)

can (Seal, Crawl)

can (Seal, Swim)

has (Mammal, Warm-blood)

ດັ່ງນັ້ນການສະແດງຄວາມຮູ້ດ້ວຍ Sematic Network ສາມາດຂະຫຍາຍໄດ້ດັ່ງຮູບທີ 3.5



ຮູບທີ 3.4 ການຂະຫຍາຍຄຸນສົມບັດເພີ່ມເຕີມຂອງ Sematic Network

Sematic Network ມີລັກສະນະພິເສດ 2 ຢ່າງຄື

1. Transitivity ໝາຍເຖິງຄວາມສາມາດໃນການຖ່າຍທອດຄວາມສຳພັນຂອງຄວາມສຳພັນທີ່ເປັນ is-a ແລະ has-part

   - is-a (X,Y)^is-a(Y,Z)→is-a(X,Z)

   is-a (Elephant, Mammal)^is-a (Mammal, Vertebrate)→is-a (Elephant, Vertebrate)

   - has-part (X, Y)^ has-part(Y,Z)→has-part(X,Z)

   has-part (Body, Hand)^has-part(Hand, Finger)→has-part (Body,Finger)

2. ການສືບທອດຄຸນສົມບັດ (Property inheritance) ຄືຂະບວນການຂອງໂນດ ຫຼື Object ຢູ່ໃນລະດັບຕ່ຳກວ່າສາມາດໃຊ້ຄວາມສຳພັນຂອງໂນດທີ່ຢູ່ສູງກວ່າໄດ້ເຊັ່ນ:

   Is-a (Elephant, Mammal)

   Do(Mammal, Giving-Milk)

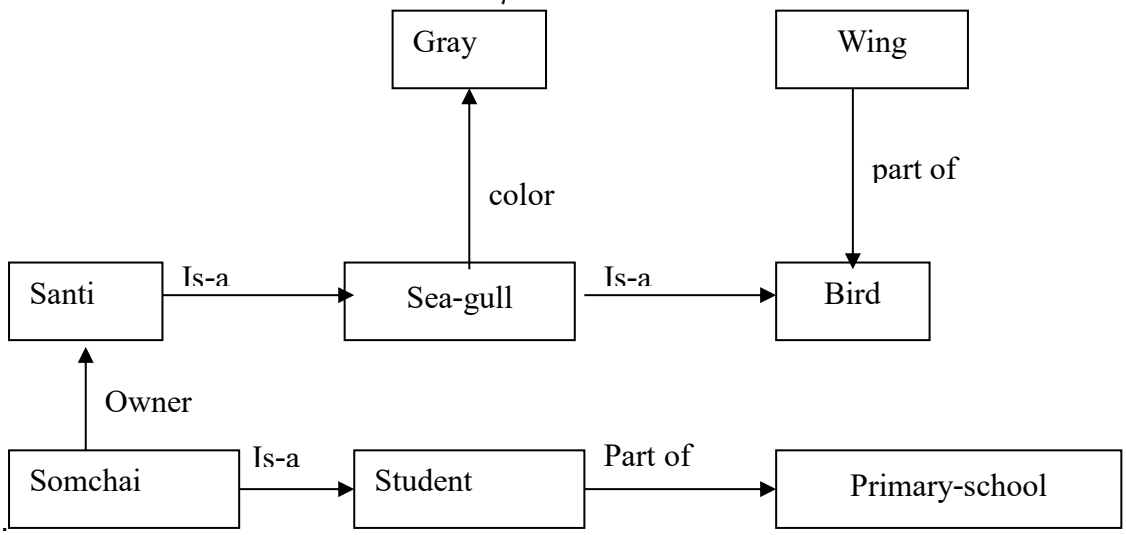
   ຈະໄດ້ Do(Elephant, Giving-Milk)

3.1.2. ການຫາເຫດຜົນດ້ວຍວິທີຂອງ Semantic Network

ການຫາເຫດຜົນຂອງຄວາມຮູ້ແບບນີ້ສາມາດເຮັດໄດ້ງ່າຍໆ ໂດຍການຄົ້ນຫາຈາກເສັ້ນເຊື່ອມ ຫຼື ອາກຂອງຖານຄວາມຮູ້ນັ້ນ ແລ້ວສ້າງການປຽບທຽບຈາກໂນດທີ່ຄືກັນເພື່ອປຽບທຽບຫາຄຳຕອບ. ຫຼື ເວົ້າ

ໄດ້ອີກແບບໜຶ່ງວ່າ ການຫາເຫດຜົນຂອງ Semantic Network ອາໄສຫຼັກການພື້ນຖານຂອງການປຽບທຽບຮູບແບບ (Pattern Matching) ແລະ ການຖ່າຍທອດຄຸນສົມບັດ (Property Inheritance)

ຕົວຢ່າງ ກຣາບສະແດງຄວາມຮູ້ແບບ sematic network ຮູບທີ 3.6 ຖ້າເຮົາຄ້ອງການຮູ້ວ່າ Santi ເປັນສີຫຍັງ, ການຫາເຫດຜົນຈະເລີ່ມຈາກໂນດທີ່ເປັນ Santi ກ່ອນ ເຮົາຈະເຫັນວ່າ Santi ມີເສັ້ນເຊື່ອມຕໍ່ is-a ທີ່ເຊື່ອມເຂົ້າກັບໂນດ Seagull ທີ່ມີ color ເປັນເສັ້ນເຊື່ອມຕໍ່ ແລະ ເຮັດໃຫ້ຮູ້ວ່າ Santi ມີສີ gray. ເນື່ອງຈາກວ່າ Santi ມີ ອາກທີ່ເປັນ is-a ຂອງ seagull ດັ່ງນັ້ນໂນດ Santi ຈຶ່ງໃຊ້ອາກຂອງ color ເພື່ອບອກສີຂອງ Santi ໄດ້ຕາມກົດການສືບທອດຄຸນສົມບັດ



ຮູບທີ 3.6 ກຣາບສະແດງຄວາມຮູ້ແບບ Sematic Network

ເພື່ອໃຫ້ເຂົ້າໃຈຫຼາຍຂຶ້ນເຮົາຈະໃຊ້ ເທີໂບໂປລ້ອກ(Turbo Prolog) ຂຽນເພື່ອສະແດງວິທີຫາເຫດຜົນ

```
/*  Program Find what is the color of Santi*/
Predicates
    Is-a(string,string)
    Owner(string,string)
    Partof(string,string)
        Color(string,string)
    Goal
    Write("Question:  What is a colorof SANTI", nl.
    Color(santi,x)
   Write("Anwer:  The color of SANTI is",x), nl.
Clauses
    Isa(santi,sea_gull).
```

Isa(sea_gull,bird).

Isa(Marcus,sea_gull).

Isa(somchai,student).

Isa(x,y) if isa(x,z) and isa(z,y)..

Color(seal_gull,gray).

Color(x,y) if isa(x,z) and isa(z,y).

Part of(wing,bird).

Part of(student,primary_school).

Part of(x,y) if part of(x,z) and isa(y,z).

Owner(somchai,santi).

ເມື່ອ compile ໂປຣແກຣມແລ້ວ Run ຈະໄດ້ຜົນຮັບດັ່ງນີ້

A:\>Santi

Question: What is a color of Santi

Answer: The color of Santi is gray

3.2. Frame

ວິທີການສະແດງຄວາມຮູບແບບ ເຟຣມ (Frame) ຖືກພັດທະນາໂດຍ ມິ້ນສະກີ (Minsky, 1975) ເປັນການສະແດງຄວາມຮູ້ອີກແບບໜຶ່ງທີ່ຂະຫຍາຍມາຈາກ sematic network ໂດຍການເພີ່ມຂ່າວສານ (information) ຂອງໂນດໃຫ້ສາມາດບັນຈຸຂໍ້ຄວາມໄດ້ຫຼາຍຂຶ້ນແທນທີ່ຈະມີພຽງ ອອບເຈັກເທົ່ານັ້ນ, ໂດຍການສະແດງຄວາມຮູ້ແບບນີ້ໃຊ້ເຟຣມໃນການອະທິບາຍເຫດການ (Event), ການເຮັດວຽກ (Action) ແລະ ສະຖານະ(Status) ຂອງວັດຖຸ (Object). ເຊິ່ງມີໂຄງສ້າງຂໍ້ມູນດັ່ງນີ້:

<Frame> ::= <Frame Name>
<Slot Name> <Slot Value>
...
<Slot Name> <Slot Value>

ຕົວຢ່າງການສະແດງຄວາມຮູ້ແບບເຟຣມ

Mammal
subclass: Animal
warm-blooded: yes
Elephant

subclass: Mammal

* color: gray
* size: large

Pi-Yai

instance: Elephant

color: white

owner: Prannoi

Pi-Noi

instance: Elephant

size: samll

ຈາກຕົວຢ່າງ <Slot Name> ທີ່ຊື່ subclass ແລະ instance ເປັນໂຕເຊື່ອມເຟຣມທັງໝົດເຂົ້າຫາກັນ, ເຟຣມ Mammal ຈະເປັນເຟຣມທີ່ຢູ່ເທິງສຸດ, ມີເຟຣມ Elephant ເປັນ class ຍ່ອຍ ຫຼື class ຍ່ອຍ ຂອງ Mammal ແລະ ເຟຣມ Elephant ມີເຟຣມທີ່ມີຊື່ Pi-Yai ແລະ ເຟຣມ Pi-Noi ເປັນເຟຣມລູກ.

3.2.1. ການຖ່າຍທອດຄຸນສົມບັດ

ການຖ່າຍທອດຄຸນສົມບັດ (Property Inheritance) ເປັນວິທີການທີ່ເຟຣມທີ່ຢູ່ໃນລະດັບຕ່ຳກວ່າ (Subclass) ສາມາດໃຊ້ຄຸນສົມບັດທີ່ຢູ່ໃນເຟຣມສູງກວ່າໄດ້. ເປັນຫຼັກການດຽວກັບການຖ່າຍທອດຄຸນສົມບັດຂອງ Sematic network, ຈາກຕົວຢ່າງ ເຟຣມ Mammal, Elephant, Pi-Yai ແລະ Pi-Noi ຂ້າງເທິງຈະຮູ້ໄດ້ວ່າ Pi-Yai

ເປັນຊ້າງສີຂາວ (ບໍ່ແມ່ນສີເທົາ), ມີຂະໜາດໃຫຍ່ ແລະ ມີເຈົ້າຂອງແມ່ນ Prannoi. ແລະເຫັນໄດ້ວ່າເຟຣມ Pi-Yai ຮັບຄຸນສົມບັດເລື່ອງຂອງຂະໜາດມາຈາກເຟຣມແມ່ຂອງມັນ (Elephant) ແຕ່ເລື່ອງສີຂາວແມ່ນເປັນຂອງເຟຣມ Pi-Yai ເອງ. ໃນທາງດຽວກັນ ເຟຣມ Pi-Noi ມີສີເທົາ ແລະ ມີຂະໜາດນ້ອຍ; ຍ້ອນເຫດຜົນດຽວກັນກັບເຟຣມ Pi-Yai ການຖ່າຍທອດຄຸນສົມບັດທີ່ຊັບຊ້ອນຂຶ້ນຄື ສິ່ງທີ່ອອບເຈັກໜຶ່ງມີ Superclass ຫຼາຍກວ່າໜຶ່ງ ແລະ ການຖ່າຍທອດຄຸນສົມບັດເປັນເລື່ອງຂອງການພິຈາລະນາວ່າຄຸນສົມບັດຂອງເຟຣມໃດທີ່ເໝາະສົມກັບ class ທີ່ກຳລັງພິຈາລະນາຢູ່ດັ່ງນີ້:

Animal

has-part: head

Mammal

subclass: Animal

has-part: nose

warm-blooded: yes

Elephant

Subclass: Mammal

has-part: trunk

* color : gray

* size: large

Pi-Yai

instance: Elephant

color: white

owner: Prannoi

ຈາກຕົວຢ່າງຂ້າງເທິງ ເຟຣມ Pi-Yai ເປັນ class ຍ່ອຍ ຂອງ Elephant ແລະ Elephant ເປັນ class ຍ່ອຍຂອງ Mammal ດັ່ງນັ້ນ Pi-Yai ຈຶ່ງຮັບຄຸນສົມບັດຂອງ Elephant ຄື ເປັນຊ້າງສີຂາວ, ມີງວງ (Trunk) ແທນມີຈະຮັບຄຸນສົມບັດຂອງ Mammal ທີ່ມີແຕ່ດັ່ງ ທຳມະດາ, ເຊິ່ງໃນກໍລະນີນີ້ກົນໄກການເລືອກຄືການເລືອກ class ທີ່ຢູ່ຕິດກັນ, ແທນການເລືອກຄຸນສົມບັດຂອງ class ທີ່ຢູ່ຫ່າງອອກໄປ

ໃນບາງກໍລະນີ ສະລັອດ (Slot) ສາມາດຮັບການຖ່າຍທອດຄຸນສົມບັດໄດ້ຫຼາຍກວ່າ 1 ຄ່າ, ເພື່ອປະຫຍັດການໃຊ້ພື້ນທີ່ເຟຣມ, ດັ່ງໃນຕົວຢ່າງຂ້າງເທິງ; ສຳລັບສະລັອດ has-part ຂອງ Mammal ທີ່ມີ Nose ຢູ່ແລ້ວ, ແຕ່ສະລັອດ has-part ຂອງ Animal ກໍ່ມີ has-part ທີ່ເປັນ head ຢູ່ຄືກັນ, ດັ່ງນັ້ນ has-part ຂອງ Mammalຄວນຈະໄດ້ຮັບຄ່າ Headເຂົ້າມານຳ, ເຊິ່ງໃນຕົວຈິງ has-part ຂອງເຟຣມ Mammal ຈະຕ້ອງລະບຸວ່າເປັນຫຼາຍຄ່າ (Multi Value). ໃນທາງດຽວກັນສະລັອດບາງອັນກໍ່ບໍ່ສາມາດຮັບຫຼາຍຄ່າ ເຊັ່ນ: ສະລັອດ color ຂອງ Elephant ແລະ Pi-Yai ຄວນເປັນຄ່າດຽວ (Single Value).

3.2.2. Slot

ການຂະຫຍາຍສະລັອດ (Slot) ແລະການສະແດງຄວາຮູ້ແບບເຟຣມນີ້, ສາມາດຂະຫຍາຍຄຸນສົມບັດຂອງສະລັອດໄດ້ ແລະ ມີໂຄງສ້າງຂໍ້ມູນດັ່ງນີ້:

<Slot> ::= <Slot Name>

<Slot Name> <Slot Value>

<Slot> ::= <Slot name>

<Facet Name> <Facet Value>
<Facet Name> <Facet Value>

……

<Slot> ::= <Slot name>

<Facet Name> <Facet Value>
<Facet Name> <Facet Value>

ໃນການຂະຫຍາຍ <Slot> ນີ້ຈະເກີດ Attribute ໃໝ່ທີ່ມີຊື່ວ່າ ຟາເຊັດ (Facet) ເຊິ່ງສາມາດແທນດ້ວຍຄຸນສົມບັດ 2 ແບບ ຄື:

1. ຟາເຊັດ (Facet) ເປັນວິທີການສະແດງຄວາມຮູ້ແບບກຳນົດຄ່າຕ່າງໆໃນເຟຣມເຊັ່ນ: Body, Window< Rang ຫຼື Default ເປັນຕົ້ນ, ດັ່ງຕົວຢ່າງລຸ່ມນີ້

   Frame: CAR

   Specialization of: LAND VEHICLE

   Body: steel

   Window: glass

   Mobility: self-propelled

   Mobility Mechanism: has wheels

   Tires: Rubber

   Model:

   Range : (sedan, convertible, 2-door, station wagon)

   Default: sedan

   Fuel:

   Range: (gasoline, diesel, propane)

   Default: gasoline

   Number of seats:

   Range: (1-9)

   Defult: none

2. Attached Procedure ເປັນຂະບວນການສະແດງຄວາມຮູ້ໂດຍ Procedure, ໂດຍທີ່ເຮົາສາມາດເພີ່ມ Procedure ເຂົ້າໄປເພື່ອຊ່ວຍໃນການເຮັດວຽກຂອງເຟຣມໃນກໍລະນີທີ່ຕ້ອງການຮູ້ຄ່າເພີ່ມເຕີມ, ແຕ່ບໍ່ມີການສະແດງຄວາມຮູ້ຢູ່ໃນເຟຣມນັ້ນ. ຂະບວນການທີ່ເພີ່ມຂຶ້ນອາໄສຄຳສັ່ງ ເຊັ່ນ: if-needed Facet, if-added Facet ແລະ if-remove Facet
   - if-needed ເປັນ Procedure ທີ່ໃຊ້ຄົ້ນຫາຄ່າໃຫ້ກັບສະລັອດເມື່ອຕ້ອງການຮູ້ຄ່ານັ້ນ
   - if-added ເປັນ Procedureທີ່ໃຊ້ສຳລັບຄິດໄລ່ຫາຄ່າທີ່ຕ້ອງການເມື່ອມີຄວາມຈຳເປັນ
   - if-remove ເຮັດໜ້າທີ່ກົງກັນຂ້າມກັບ if-added ຄືການລຶບຂໍ້ມູນອອກເມື່ອຈຳເປັນ

ຕົວຢ່າງຕໍ່ໄປນີ້ເປັນການສະແດງຄວາມຮູ້ດ້ວຍເຟຣມທີ່ມີການໃຊ້ Attached Procedure ຈະເຫັນໄດ້ວ່າໃນສະລັອດຂອງ Fuel Remaining ທີ່ມີຟາເຊັດເປັນ Range ເຊິ່ງຈະບອກລະດັບນ້ຳມັນຂອງລົດທຸກລະດັບທີ່ຖືກຕ້ອງເມື່ອເວລາຜ່ານໄປ, ແຕ່ບໍ່ຮູ້ວ່າປັດຈຸບັນມີນ້ຳມັນຢູ່ທ່ໍໃດ. ໃນກໍລະນີທີ່ຕ້ອງການຮູ້ວ່າ

ປັດຈຸບັນມີນ້ຳມັນໃນຖັງເຫຼືອທ່ໃດ ກ່ສາມາດຮູ້ໄດ້ໂດຍການໃຊ້ if-needed ທີ່ມີ Procedure: check fuel gauge ໄປກວດສອບແລ້ວນຳຄ່າທີ່ໄດ້ເກັບໄວ້ໃນ Facet: Value

ສຳລັບ If-needed ຈະເຮັດວຽກເມື່ອມີການຖາມຫາວ່າລົດຄັນນີ້ອາຍຸທ່ໃດ ຟາເຊັດທີ່ເປັນ year ຈະບອກຕ່າປີຂອງປັດຈຸບັນ, ຈາກນັ້ນ If-needed: compute years ຈະເປັນໂຕຄິດໄລ່ໂດຍການນຳປີປັດຈຸບັນໄປລົບປີທີ່ຜະລິດລົດ ຄື ຟາເຊັດ year of Manufactured ເຊິ່ງມີຢູ່ໃນເຟຣມອອກມາເປັນອາຍຸຂອງລົດ, ແລ້ວຄ່າທີ່ໄດ້ເກັບໄວ້ໃນ <value> ທີ່ຢູ່ທາງລຸ່ມ

Frame: CAR

Specialization of: LAND VEHICLE

Body: Steel

Window: Glass

Fuel Remaining:

Range : (empty, ¼ tank, ½ tank, full)

Default : none

IF-NEEDED: check fuel gauge

Year of Manufactured:

Year: 1987

Default: none

IF-ADDED: compute years

Value:

ຕົວຢ່າງຂອງ Attached Procedureໃນນີ້ຄື IF-NEEDED IF-ADDED

3.3. ການເພິ່ງພາດ້ານມະໂນພາບ (Conceptual dependency)

ການເພິ່ງພາດ້ານມະໂນພາບ (Conceptual dependency) ຫຼື CD ເປັນທິດສະດີຂອງພາສາທຳມະຊາດ (Natural Language)ແລະ ການປະມວນຜົນພາສາທຳມະຊາດ (Natural Language Processing) ເຊິ່ງຖືກພັດທະນາໂດຍແຊງ (Schank, 1972) ດ້ວຍຈຸດປະສົງທີ່ຕ້ອງການສ້າງໂປຣແກຣມຄອມພິວເຕີໃຫ້ສາມາດເຂົ້າໃຈພາສາມະນຸດໄດ້ດີ ແລະ ສະຫຼຸບເນື້ອໃນອອກມາແປເປັນພາສາອື່ນໄດ້ ແລ້ວຍັງສາມາດຕອບຄຳຖາມກ່ຽວກັບເລື່ອງເຫຼົ່ານັ້ນໄດ້ເຊັ່ນ: ຖ້າເວົ້າວ່າ “Somchai sold his car.” ຈະເຫັນວ່າໃນປະໂຫຍກນີ້ມີສິ່ງທີ່ເຊື່ອງຢູ່ພາຍໃນຄືເງິນ, ເມື່ອຄົນອ່ານແລ້ວຈະຮູ້ວ່າສິ່ງນີ້ຈະເປັນສິ່ງທີ່ກ່ຽວຂ້ອງຢູ່ພາຍໃນ, ເຊິ່ງບ່ໄດ້ມີການເວົ້າເຖິງເລີຍ ຫຼື ຖ້າມີໃຜຖາມວ່າ ດຽວນີ້ສົມຊາຍເປັນເຈົ້າຂອງລົດບ່? ເຄື່ອງຄອມພິວເຕີໃດກ່ຕາມທີ່ສາມາດເຂົ້າໃຈພາສາມະນຸດໄດ້ດີຈະຕ້ອງຕອບວ່າ “ບ່” ແລະຈະຕ້ອງຕອບວ່າ “ສົມຊາຍໄດ້ຮັບເງິນ” ຖ້າມີການຖາມວ່າ “ສົມຊາຍໄດ້ຮັບເງິນຫຼືບ່”

ສິ່ງທີ່ເຮັດໃຫ້ຄອມພິວເຕີສາມາດເຮັດໄດ້ຄືກັບສິ່ງທີ່ກ່າວມາຂ້າງເທິງ, ຜູ້ອອກແບບລະບົບຈະນຳເອົາຄວາມຮູ້ໃສ່ໃນຄອມພິວເຕີຈະຕ້ອງມີຄວາມເຂົ້າໃຈເປັນຢ່າງດີກ່ຽວກັບວິທີການຫາຂໍ້ສະຫຼຸບຄວາມຮູ້ຂອງມະນຸດ ແລະ ຕ້ອງມີຄວາມເຂົ້າໃຈກ່ຽວກັບວິທີການສະແດງຄວາມຮູ້. ເຊິ່ງການເພິ່ງພາດ້ານມະໂນພາບເປັນທິດສະດີທີ່ອະທິບາຍວິທີການສະແດງຄວາມຮູ້ໃນທາງຂອງຄວາມໝາຍ.

3.3.1. ກົດຂອງການເພິ່ງພາດ້ານມະໂນພາບ

ກົດຂອງການເພິ່ງພາດ້ານມະໂນພາບເປັນການສະສົມແນວຄວາມຄິດຕ່າງໆ, ທີ່ເປັນເລື່ອງຂອງປະສົບການດັ່ງໃນຕົວຢ່າງທີ່ກ່າວມາຂ້າງເທິງ “Somchai sole his car” ຈາກປະສົບການຈະສາມາດເຂົ້າໃຈໄດ້ວ່າເງິນຈະຕ້ອງເຂົ້າມາກ່ຽວຂ້ອງ ແລະ ກົດຂອງ CD ກໍ່ຈະສະສົມແນວຄວາມຄິດກ່ຽວກັບເລື່ອງເຫຼົ່ານີ້ໄວ້, ໃນການເວົ້າເຖິງເລື່ອງເຫຼົ່ານີ້ສິ່ງທີ່ຈະຕ້ອງອ້າງເຖິງສະເໝີຄື Sematic, ເຊິ່ງໃນຄວາມໝາຍຂອງ CD ຈະໝາຍເຖິງ ກົດຂອງປະສົບການທີ່ໃຊ້ສຳລັບການສະສົມແນວຄວາມຄິດ. ເປົ້າໝາຍຂອງການສະແດງຄວາມຮູ້ໃນລັກສະນະນີ້ຄື:

- ການສະແດງຄວາມຮູ້ເລີ່ມຈາກປະໂຫຍກ
- ເພື່ອເຮັດໃຫ້ປະໂຫຍກສາມາດຫາຂໍ້ສະຫຼຸບໄດ້
- ເພື່ອເຮັດໃຫ້ຄຳທີ່ໃຊ້ໃນປະໂຫຍກນຳເຂົ້າກັບຄຳທີ່ໃຊ້ສະແດງຄວາມຮູ້ມີຄວາມເຊື່ອມເຂົ້າກັນໂດຍກົງ
- ເພື່ອກຳນົດປະໂຫຍກທີ່ຂຽນຕ່າງກັນແຕ່ມີຄວາມໝາຍຄືກັນ, ມີວິທີການສະແດງຄວາມຮູ້ເປັນຮູບແບບດຽວກັນ

❖ ວິທີການສະແດງຄວາມຮູ້ຂອງ CD ມີຫຼັກການດັ່ງນີ້:

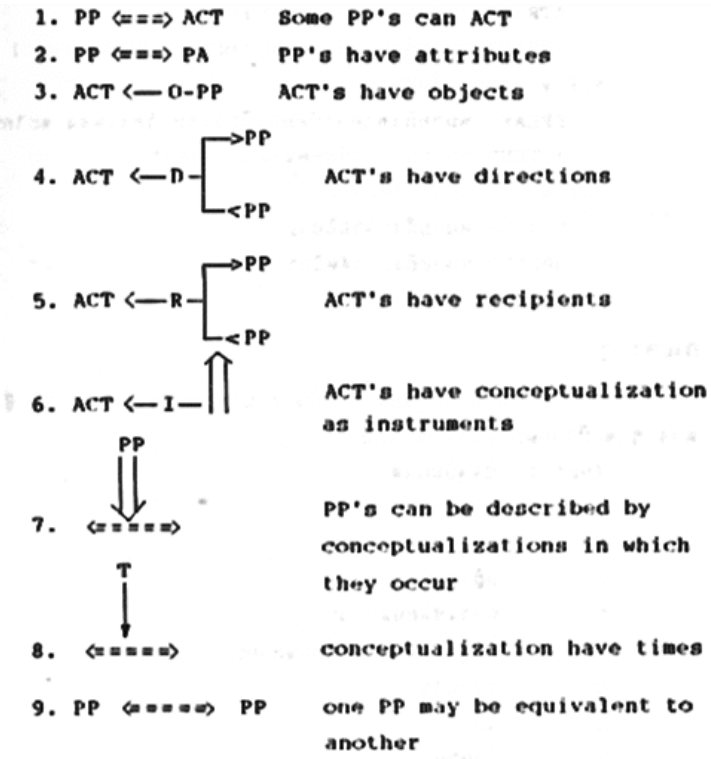
- ມີການກຳນົດຮູບແບບຂອງໂນດທີ່ລະບຸວິທີການຂຽນຂໍ້ມູນທີ່ຈະແຈ້ງລົງໃນໂຄງສ້າງທີ່ກຳນົດໃຫ້
- ມີຊຸດຄຳສັ່ງມາດຕະຖານ (Primitive) ທີ່ຈະແຈ້ງ
- ມີການກຳນົດວິທີປະກອບໂຄງສ້າງຂອງການສະແດງຄວາມຮູ້

❖ ຊຸດຄຳສັ່ງມາດຕະຖານທີ່ບອກເຖິງຊະນິດຂອງວັດຖຸທີ່ໃຊ້ໃນກົດຂອງ CD ມີດັ່ງຕໍ່ໄປນີ້:

- PPs; Picture Procedures: Object ຕ່າງໆ ວັດຖຸທາງກາຍຍະພາບ: PPs ອາດເປັນຜູ້ເຮັດວຽກ ແລະ ຜູ້ຖືກເຮັດວຽກ (ເຊັ່ນ: ຄົນ, ສິ່ງຂອງ ເປັນຕົ້ນ) ບາງຄັ້ງອາດເປັນຈຸດເລີ່ມຕົ້ນ ຫຼື ຈຸດໝາຍປາຍທາງກໍ່ໄດ້. ວັດຖຸທາງມະໂນພາບ: PPs ຈະເປັນຄຸນສົມບັດບາງຢ່າງກໍ່ໄດ້ ເຊັ່ນ ຄວາມຮ້ອນ
- PAs; attributes of PP’s state(Value) ໝາຍເຖິງຄ່າທີ່ບອກລະດັບຂອງ Attribute ເຊັ່ນ ຟ້າ ເປັນລັກສະນະຂອງ Attribute ຂອງສີ ແລະ 5 ແມັດ ເປັນລັກສະນະຂອງ Attribute ຂອງຄວາມສູງ
- ACTs; action ໝາຍເຖິງການເຮັດວຽກຕໍ່ Object ໂດຍຜູ້ເຮັດວຽກ(PP)
- AAs; Action Aiders ໝາຍເຖິງສິ່ງທີ່ຂະຫຍາຍເພີ່ມເຕີມຊະນິດຂອງ ACT
- LOCs; Location ໝາຍເຖິງສະຖານທີ່
- Ts; time ໝາຍເຖິງເວລາ

3.3.2 ກົດຕ່າງໆ ແລະ ໂຄງສ້າງຂອງ CD

ໃນ CD ມີກົດມາດຕະຖານທີ່ຖືກອອກແບບໂດຍແຊງ (Schank, R.C., 1975) ເພື່ອອະທິບາຍແນວຄິດດ້ານມະໂນພາບຂອງການເຮັດວຽກໃນຮູບແບບຕ່າງໆໄວ້ ດັ່ງສະແດງໃນຮູບທີ 3.7



ຮູບທີ 3.7 ກົດແລະໂຄງສ້າງຂອງ CD

3.3.3 ACT ຊະນິດຕ່າງໆໃນ CD

ການເຮັດວຽກ ຫຼື ACT ຂອງການສະແດງຄວາມຮູ້ແບບເພິ່ງພາດ້ານມະໂນພາບຂອງການເຮັດວຽກເມື່ອມີການສະແດງອອກເປັນອາການ (Physical Action) ມີ 5 ຊະນິດດັ່ງນີ້:

- MOVE ໝາຍເຖິງການເຄື່ອນໄຫວສ່ວນຕ່າງໆ ຈອງຮ່າງກາຍ
- PROPEL ໝາຍເຖິງການຍ້າຍວັດຖຸທີ່ເກີດຈາກຜູ້ເຮັດວຽກ
- INGEST ໝາຍເຖິງການນຳເອົາສິ່ງຂອງເຂົ້າຈາກຮ່າງກາຍ
- EXPEL ໝາຍເຖິງການນຳເອົາສິ່ງຂອງອອກຈາກຮ່າງກາຍ
- GRASP ໝາຍເຖິງການຈັບ ຫຼື ຖືວັດຖຸ

- ACT ສຳລັບການປ່ຽນແປງສະຖານະ
  - ATRANS ໝາຍເຖິງການຖ່າຍທອດທາງວັດຖຸ
  - PTRANS ໝາຍເຖິງການຖ່າຍທອດທາງນາມມະທຳ (Abstract)
- ACT ສຳລັບການສື່ສານ
  - SPEAK ໝາຍເຖິງອາການທີ່ອອກສຽງ ເຊັ່ນ ຮ້ອງເພງ ຮ້ອງສຽງດັງ
  - ATTEND ໝາຍເຖິງການທີ່ບຸກຄົນໄດ້ຮັບຂ່າວສານ
- ACT ສຳລັບສະໝອງ (Mental)

- MTRANS ໝາຍເຖິງການອະທິບາຍ
- BUILD ໝາຍເຖິງການຕັດສິນໃຈ

3.3.4 ການລະຕ່າງໆ

ໃນການສະແດງຄວາມຮູ້ຂອງ CD ນັ້ນສາມາດລະບຸການລະຕ່າງໆ ໄດ້ ເຊິ່ງການລະຕ່າງໆ ຈະມີລາຍລະອຽດ ດັ່ງຕໍ່ໄປນີ້:

(null) ປັດຈຸບັນ

p ອະດີດ

f ອະນາຄົດ

/ ປະຕິເສດ

ts ການປ່ຽນສະຖານະ

tf ສິ້ນສຸດການປ່ຽນສະຖານະ

c ເງື່ອນໄຂ

k ຕໍ່ເນື່ອງ

? ຄຳຖາມ

3.3.5 ວິທີການສະແດງປະໂຫຍກທາງພາສາສາດ ດ້ວຍການເພິ່ງພາດ້ານມະໂນພາບ

1. ແຍກອົງປະກອບຂອງປະໂຫຍກ ເຊັ່ນ ACTOR, TIME, ACTION, OBJECTS ແລະ DIRECTION.
2. ຫາຄຳທີ່ໃຊ້ແທນຊື່ຂອງ ACTION ນັ້ນເຊັ່ນ ການຍ້າຍວັດຖຸ(ໂດຍຄົນ) ໃນທາງການເພິ່ງພາເຊິງມະໂນພາບໃຊ້ຄຳວ່າ PROPEL
3. ພິຈາລະນາຮູບແບບຂອງກົດ ແລະ ການຂຽນຮູບ CD ຂຶ້ນມາ

<u>ຕົວຢ່າງການຂຽນ CD ເຊັ່ນ:</u>

1. ແຍກອົງປະກອບຂອງປະໂຫຍກ

   ACTOR: John

   TIME: present

   ACTION: moves

   OBJECT: his hand

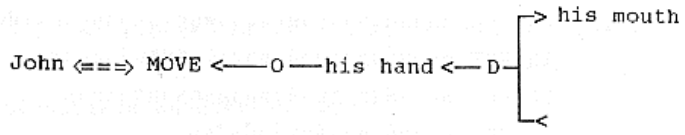
   DIRECTION: FROM: ບໍ່ມີ (ເນື່ອງຈາກບໍ່ຮູ້ວ່າຢູ່ຕຳແໜ່ງໃດ)

   TO: his mouth

2. ຫາຄຳທີ່ໃຊ້ແທນທີ່ຂອງ ACTION ນັ້ນ

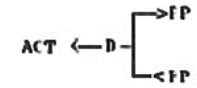
ໃນທີ່ນີ້ action ຄື moves ເຊິ່ງຄືກັບໃນ CD ຄື MOVE

3. ພິຈາລະນາຮູບແບບຂອງກົດ ແລະ ການຂຽນຮູບ CD ຂຶ້ນມາ



ໂດຍໃຊ້ກົດທີ່1 PP⇔ACT

ໂດຍໃຊ້ກົດທີ່ 3 $ACT \leftarrow^{0} PP$

ໂດຍໃຊ້ກົດທີ່ 4 

ຂໍ້ດີຂອງການສະແດງຄວາມຮູ້ແບບເພິ່ງພາດ້ານມະໂນພາບຄືການຫຼຸດຂັ້ນຕອນການຫາຂໍ້ສະຫຼຸບ. ເພາະໃນການສະແດງຄວາມຮູ້ແບບ CD ຈະເປັນໂຄງສ້າງທີ່ກຳນົດຈາກການຫາຂໍ້ສະຫຼຸບໄວ້ລ່ວງໜ້າແລ້ວ. ແຕ່ຈຸດດ້ອຍແມ່ນວິທີການສະແດງຄວາມຮູ້ແບບນີ້ຊັບຊ້ອນຫຼາຍ ແລະ ການກຳນົດຄຳສັ່ງມາດຕະຖານໃຫ້ຄົບຖ້ວນກັບການສະແດງຄວາມຮູ້ທີ່ຫຼາກຫຼາຍເຮັດໄດ້ຍາກຫຼາຍ.

ບົດທີ 4: ການຫາຂໍ້ສະຫຼຸບພາຍໄຕ້ຄວາມບໍ່ແນ່ນອນ (Inference under Uncertainty)

ການຮຽນຮູ້ ແລະ ການຫາເຫດຜົນໃນປັນຍາປະດິດຈະຕ້ອງພົບກັບຄວາມຮູ້ຕ່າງໆຢ່າງຫຼວງຫຼາຍ, ທີ່ເປັນຄວາມຮູ້ແບບລຽບງ່າຍໃຈຄວາມສົມບູນ ແລະ ຄວາມຮູ້ທີ່ມີຄວາມຊັບຊ້ອນຈົນຕ້ອງອາໄສປະສົບການເພື່ອສ້າງຄວາມເຂົ້າໃຈສຳລັບຄວາມຮູ້ ແລະ ຂໍ້ມູນທີ່ມີຄວາມຈະແຈ້ງຈະສະດວກໃນການນຳໄປໃຊ້.

4.1 Uncertainty

ຄວາມຮູ້ທີ່ຜ່ານມາສ່ວນຫຼາຍເປັນຄວາມຮູ້ທີ່ມີໃຈຄວາມສົມບູນ ແລະ ເນື້ອໃນຈະແຈ້ງສາມາດຫາຂໍ້ສະຫຼຸບໄດ້ຈາກການຄົ້ນຫາຄ່າຄວາມຈິງ, ແຕ່ຕົວຈິງແລ້ວປະໂຫຍກ ຫຼື ເນື້ອໃນບາງສ່ວນໃນອີກທາງໜຶ່ງ ເຫັນວ່າຂາດຄວາມຈະແຈ້ງ ແລະ ມີຄວາມຊ້ຳຊ້ອນ ເຮັດໃຫ້ເກີດການສົງໃສ ແລະ ບໍ່ໝັ້ນໃຈ ສິ່ງທີ່ເກີດຂຶ້ນຄືດັ່ງນັ້ນຈຶ່ງເອີ້ນວ່າ: "**ຄວາມບໍ່ແນ່ນອນ (Uncertainty)**".

ຄວາມບໍ່ແນ່ນອນ ໝາຍເຖິງສະຖານະການຄວາມລັງເລ, ສົງໃສ, ບໍ່ແນ່ໃຈ ຫຼື ບໍ່ໝັ້ນໃຈ ສົ່ງຜົນໃຫ້ສະຖານະການເກີດຄວາມບໍ່ໝັ້ນຄົງ ແລະ ຜົນຮັບບໍ່ໄດ້ຄວາມແນ່ນອນ ເນື່ອງຈາກຂາດປັດໃຈທີ່ຈະຮັບປະກັນຄວາມຖືກຕ້ອງ ແລະ ສົມບູນ. ຄວາມບໍ່ແນ່ນອນເຫຼົ່ານີ້ອາດເກີດໄດ້ຈາກຫຼາຍສາເຫດຂຶ້ນຢູ່ກັບສະຖານະການຕ່າງໆດເຊັ່ນ: ການສຳຫຼວດ ແບບໃຊ້ແບບສອບຖາມ ເຊິ່ງຜູ້ຕອບແບບສອບຖາມແຕ່ລະຄົນອາດມີຄວາມເຫັນບໍ່ກົງກັນ, ໃນບາງສ່ວນອາດມີຄວາມເຫັນແຕກຕ່າງຈາກຄົນສ່ວນຫຼາຍໂດຍສິ້ນເຊິງເຮັດໃຫ້ມີຄ່າຄວາມບໍ່ແນ່ນອນເກີດຂຶ້ນ, ທັງນີ້ອາດແຕກຕ່າງກັນຕາມກຸ່ມເປົ້າໝາຍທີ່ຕອບແບບສອບຖາມດ້ວຍ. ຄວາມບໍ່ແນ່ນອນອາດເຮັດໃຫ້ເກີດການຜັນປ່ຽນຂອງຂໍ້ມູນເຊິງເກີດຈາກປັດໃຈຫຼາຍຢ່າງ ເຊັ່ນ:

- ຂໍ້ມູນບໍ່ຄົບຖ້ວນ ມີພຽງບາງສ່ວນ
- ຫຼັກຖານຢືນຢັນຄວາມໜ້າເຊື່ອຖືມີບໍ່ພຽງພໍທີ່ຈະສ້າງຄວາມຫມັ້ນໃຈໃນຂໍ້ມູນໄດ້
- ການໃຊ້ພາສາມີຮູບແບບທີ່ເຂົ້າໃຈຍາກ ຫຼືຕີຄວາມໄດ້ຍາກ
- ຂໍ້ມູນຈາກແຫຼ່ງອ້າງອີງແຕກຕ່າງກັນ ຈົນເກີດຄວາມຂັດແຍ້ງ
- ບໍ່ສາມາດຊີ້ແຈ້ງຂໍ້ມູນດ້ວຍເຫດແລະຜົນໄດ້ຢ່າງຈະແຈ້ງ
- ຂໍ້ມູນອາດໄດ້ມາຈາກການກະຕ່ອງ ເຮັດໃຫ້ເກີດຄວາມບໍ່ໝັ່ນໃຈ
- ແຫຼ່ງທີ່ມາຂອງຂໍ້ມູນຂາດຄວາມໜ້າເຊື່ອຖື

4.1.1 ການແທນຄ່າຄວາມບໍ່ແນ່ນອນ

ການແທນຄ່າຄວາມບໍ່ແນ່ນອນສາມາດເຮັດໄດ້ຫຼາຍວິທີ ເຊິ່ງແຕ່ລະວິທີຈະມີການນຳສະເໜີຄ່າຄວາມບໍ່ແນ່ນອນດ້ວຍຂໍ້ມູນຫຼາຍລັກສະນະ ວິທີການພື້ນຖານໃນການແທນຄ່າຄວາມບໍ່ແນ່ນອນມີດັ່ງນີ້

☐ ການແທນຄ່າດ້ວຍຕົວເລກ

ເປັນວິທີການທີ່ງ່າຍແລະສະດວກທີ່ສຸດ ໂດຍກຳນົດຊ່ວງຂອງຕົວເລກເຊັ່ນ 0 ແທນຄ່າຄວາມບໍ່ແນ່ນອນ ແລະເມື່ອຕົວເລກຫຼາຍຂຶ້ນສະແດງວ່າຄ່າຄວາມແນ່ນອນຫຼາຍຂຶ້ນດ້ວຍ

☐ ການແທນຄ່າດ້ວຍກຣາບແລະແຜນວາດ

ເປັນວິທີການທີ່ຊ່ວຍລຸດຄວາມບໍ່ໝັ້ນໃຈແລະຄວາມຂັດແຍ່ງທີ່ເກີດຈາກການກຳນົດລະດັບຄວາມບໍ່ແນ່ນອນດ້ວຍຕົວເລກ

☐ ການແທນຄ່າດ້ວຍສັນຍາລັກ

ເປັນການແທນຄ່າຄວາມບໍ່ແນ່ນອນໂດຍການກຳນົດການອັດ ເຊິ່ງຄ້າຍກັບການແທນຄ່າດ້ວຍກຣາບ ແຕ່ຈະມີການກຳນົດການອັດຢ່າງເປັນລະດັບຢ່າງຈະແຈ້ງ

4.1.2 ການຫາເຫດຜົນແບບ Monotonic

ໃນຂະບວນການທາງປັນຍາປະດິດການຫາເຫດຜົນຈະສ້າງດ້ວຍການນຳສະເໜີຄວາມຮູ້ຕ່າງໆ ທີ່ເຄື່ອງຄອມພິວເຕີບໍ່ເຂົ້າໃຈ ຫຼື ສາມາດວິເຄາະ ແລະ ສະຫຼຸບຄວາມໄດ້ຢ່າງຖືກຕ້ອງ. ການຫາເຫດຜົນທີ່ມີຄວາມບໍ່ແນ່ນອນຈຳເປັນຕ້ອງນຳຂະບວບການທາງຕັກກະສາດເຂົ້າມາຊ່ວຍ ເພື່ອໃຫ້ໄດ້ຄວາມຮູ້ທີ່ມີຄວາມຫັດກຸ່ມ ແລະ ກົງກັບຄວາມຈິງທີ່ສຸດ ໃນຂະນະທີ່ເຄື່ອງຄອມພິວເຕີກໍ່ສາມາດເຂົ້າໃຈໄດ້ເຊັ່ນກັນ. ຕັກກະສາດທີ່ມີຄວາມສຳຄັນໃນການຫາເຫດຜົນແບບນີ້ໄດ້ແກ່ Predicate Logic ໂດຍໃຊ້ຫຼັກການທີ່ວ່າ ຄວາມຈິງສາມາດກຳນົດໄດ້ ຫຼື ສ້າງຂຶ້ນມາໃໝ່ຈາກກົດຕ່າງໆ ທີ່ກົງກັບຖານຄວາມຮູ້ (Knowledge Base) ເຊິ່ງເມື່ອກຳນົດຂຶ້ນແລ້ວຄວາມຮູ້ເຫຼົ່ານີ້ຈະເປັນຈິງສະເໝີ, ແລະ ຈະບໍ່ຄັດແຍ່ງກັບສິ່ງເກົ່າທີ່ມີຢູ່ແລ້ວ. ເພື່ອເຮັດໃຫ້ຜົນຂອງຂະບວນການທາງຕັກກະສາດ ທີ່ແນ່ນອນການຫາເຫດຜົນໃນລັກສະນະນີ້ເອີ້ນວ່າ: “Monotonic”

ການຫາເຫດຜົນດ້ວຍວິທີ Monotonic ຈຳນວນຂອງຄວາມຈິ່ງພິສູດແລ້ວວ່າຖືກຕ້ອງເມື່ອຮອດໄລຍະໜຶ່ງຈະມີຄ່າຄວາມໜ້າເຊື່ອຖືເພີ່ມຂຶ້ນສະເໝີໂດຍບໍ່ຫຼຸດລົງ, ແຕ່ກໍມີຂໍ້ຈຳກັດຕ່າງໆບາງອັນເຂົ້າມາເປັນປັດໃຈສົ່ງຜົນໃຫ້ຂໍ້ມູນບໍ່ມີຄວາມສົມບູນພຽງພໍທີ່ຈະໃຊ້ຕັດສິນ ເຊິ່ງເກີດຈາກຄວາມປ່ຽນແປງຂອງຕົວປ່ຽນບາງອັນເຊັ່ນ: ເວລາ. ຂໍ້ຈຳກັດໃນເລື່ອງຄວາມສົມບູນນີ້ທີ່ເຮັດໃຫ້ວິທີການ Monotonic ບໍ່ສາມາດຫາ

ເຫດຜົນຈາກຄວາມຮູ້ໄດ້ ເຊັ່ນ “ ມື້ນີ້ອາກາດຮ້ອນຫຼາຍ “ ການບອກລະດັບຄວາມຮ້ອນຍັງອາໄສການບອກຈາກຄວາມຮູ້ສຶກ ຈຶ່ງຈຳເປັນຕ້ອງຫາວິທີບອກລະດັບຄວາມຮ້ອນ, ຫຼື ຄຳວ່າ “ ການຫຼີ້ນກິລາບານເຕະແບບບຸກຈະມີໂອກາດຊະນະຫຼາຍກ່ວາແບບຕັ້ງຮັບ “ ເຊິ່ງຂໍ້ຄວາມນີ້ບໍ່ສາມາດພິສູດຄວາມຖືກຕ້ອງໄດ້ຈະແຈ້ງ ເພາະບໍ່ສາມາດບອກລະດັບຂອງໂອກາດທີ່ຈະຊະນະໄດ້ດ້ວຍແຜນການແນ່ນອນ

ຈາກທີ່ໄດ້ກ່າວມາເຮັດໃຫ້ຕ້ອງອາໄສວິທີການແບບ ເຊິ່ງເປັນຂະບວນການສະແດງຄວາມຮູ້ທີ່ສາມາດບອກລັກສະນະຄວາມໜ້າເຊື່ອຖືອອກມາເປັນຕົວເລກໄດ້. ຈາກການຄຳນວນຫາຄ່າຄວາມເຊື່ອໝັ້ນໃຫ້ກັບຜົນຮັບທີ່ເກີດຂຶ້ນໃໝ່ໄດ້ ເຊິ່ງຄ່າເຫຼົ່ານີ້ສາມາດປ່ຽນແປງໄດ້ເມື່ອມີຂໍ້ມູນໃໝ່ເພີ່ມຂຶ້ນ ໂດຍລາຍລະອຽດຂອງຄ່າຄວາມແນ່ນອນຈະກ່າວໃນຫົວຂໍ້ຕໍ່ໄປ.

4.1.3 ຄ່າຄວາມແນ່ນອນ

ຄ່າຄວາມແນ່ນອນ (Certainty Factor: CF) ເປັນຄ່າຂອງຕົວເລກທີ່ຖືກກຳນົດຂຶ້ນບົນພື້ນຖານຂອງເຫດການ

ຕາຕະລາງທີ 4.1 ສະແດງຄ່າ CF ໃນຊ່ວງຕ່າງໆ ແລະ ຄວາມໝາຍ

Term	Certainty Factor
ບໍ່ແມ່ນຢ່າງແນ່ນອນ (Definitely not)	-1.0
ເກືອບຈະບໍ່ແມ່ນຢ່າງແນ່ນອນ (Almost certainty not)	-0.8
ບໍ່ເປັນໄປໄດ້ (Probably not)	-0.6
ອາດຈະເປັນໄປໄດ້ (Maybe not)	-0.4
ບໍ່ຮູ້ (Unkown)	-0.2 ເຖິງ +0.2
ອາດຈະແມ່ນ (Maybe)	+0.4
ເປັນໄປໄດ້ວ່າຈະແມ່ນ (Probably)	+0.6
ເກືອບຈະແມ່ນຢ່າງແນ່ນອນ (Almost certainty)	+0.8
ແມ່ນຢ່າງແນ່ນອນ (Definitely)	+1.0

CF ຖືກພັດທະນາຂຶ້ນເພື່ອໃຊ້ກັບລະບົບ MYCIN ເຊິ່ງເປັນລະບົບຜູ້ຊ່ຽວຊານ (Expert System) ສຳລັບ CF ອາດບໍ່ແມ່ນວິທີການທີ່ເປັນທາງການໃນການກຳນົດລະດັບຄ່າຄວາມບໍ່ແນ່ນອນ, ຕໍ່ການສະຫຼຸບຜົນເປັນໄປຕາມ .ການກຳນົດຄ່າ CF ຂອງກົດແຕ່ລະຂໍ້ໃນລະບົບຈະຖືກກຳນົດຂຶ້ນໂດຍຜູ້ຊ່ຽວຊານທີ່ກ່ຽວຂ້ອງກັບສະຖານະການນັ້ນໆ ສຳລັບຄ່າ CF ທີ່ໄດ້ຈະຂຶ້ນຢູ່ກັບຄວາມຮູ້ ແລະ ປະສົບການຂອງຜູ້ຊ່ຽບຊານ, ຄ່າ CF ຈະຊ່ວຍບົ່ງບອກເຖິງຄວາມໜ້າເຊື່ອຖືຂອງຂໍ້ສະຫຼຸບໄດ້ຕາມສະຖານະການ ເຊັ່ນ ລົດຄັນ

ໜຶ່ງ ມີເຄື່ອງຈັກ ແລະ ໝໍ້ໄຟໃນສະພາບປົກກະຕິ ມີນ້ຳມັນຢູ່ປະລິມານໜຶ່ງ ດັ່ງນັ້ນລົດຄັນດັ່ງກ່າວ ສະຖານະໃຊ້ງານໄດ້ ໂດຍຜູ້ຊ່ຽວຊານອາດກຳນົດໃຫ້ຄ່າ CF ເປັນ 0.8 ໃນຂະນະດຽວກັນຫາກມີຜູ້ ຊ່ຽວຊານທີ່ມີປະສົບປະການຫຼາຍກ່ວາອາດກຳນົດຄ່າ CF ເປັນໜຶ່ງ 0.6 ຕາມປັດໃຈແວດລ້ອມທີ່ໃຊ້ ສະຫຼຸບຜົນເປັນຕົ້ນ [Negnevitsky 2005].

ຄ່າ CF ໃນລະບົບຜູ້ຊ່ຽວຊານຈະນຳມາໃຊ້ຄວບຄູ່ກັບຖານຄວາມຮູ້ເຊິ່ງມີຄວາມກ່ຽວຂ້ອງກັບກົດ ຕ່າງໆ ທີ່ໃຊ້ໃນການສ້າງຖານຄວາມຮູ້ໂດຍມີໂຄງສ້າງດັ່ງນີ້.

IF <Evidence (E) >
THEN < Hypothesis (H) { CF }

ໝາຍວ່າ ທຸກໆຄ່າຂອງ CF ຈະບົ່ງບອກເຖິງຄວາມໜ້າເຊື່ອຖືໄດ້ວ່າ ເຫດການ (E)

ຈະສົ່ງຜົນໃຫ້ເກີດຂໍ້ສົມມຸດຖານ (H) ໃນທີ່ສຸດ ຄ່າ CF ບໍ່ວ່າຈະຢູ່ໃນຊ່ວງໃດກໍສາມາດຕີຄວາມໄດ້ ໂດຍອີ່ງໃສຕາຕະລາງ 4.1. ສຳລັບທິດສະດີຂອງ CF ຈະມີຄວາມກ່ຽວຂ້ອງກັບຟັງຊັນທີ່ເປັນ ມາດຕະຖານໃນການວັດແທກຄ່າ Belief ແລະ Disbelief ໄດ້ແກ່ Measure of Belief (MB(H,E)) ແລະ Measure of Disbelief (MD(H,E)) ຟັງຊັນດັ່ງກ່າວຈະໃຊ້ເປັນຕົວວັດແທກລະດັບຂອງຄວາມໜ້າ ເຊື່ອຖື, ໃນຕົວສົມມຸດຖານທີ່ກ່ຽວຂ້ອງກັບເຫດການທີ່ໜ້າສົນໃຈ ໂດຍສາມາດຫາຄ່າໄດ້ຈາກສົມຜົນດັ່ງນີ້:

$$MB(H,E) = \begin{cases} 1 & if\ p(H) = 1 \\ \dfrac{\max[p(H \mid E), p(H) - p(H)}{\max[1,0] - p(H)} & otherwise \end{cases}$$

$$MD(H,E) = \begin{cases} 1 & if\ p(H) = 0 \\ \dfrac{\min[p(H \mid E), p(H) - p(H)}{\min[1,0] - p(H)} & otherwise \end{cases}$$

ໂດຍ:

p(H) ໝາຍເຖິງ ຄ່າກະຕວງກ່ອນໜ້າທີ່ສົມມຸດຖານ H ຈະມີຄ່າເປັນຈິງ ໂດຍບໍ່ຢູ່ພາຍໃຕ້ເຫດການໃດໆ

p(H/E) ໝາຍເຖິງ ຄ່າກະຕວງຂອງສົມມຸດຖານ H ທີ່ມີຄ່າຄວາມຈິງເມື່ອຢູ່ພາຍໃຕ້ເຫດການ E

ດັ່ງນັ້ນ ການຈະຫາຄ່າຂອງ CF ຈຳເປັນຕ້ອງອາໄສຄ່າຂອງ MB(H,E) ແລະ MD(H,E) ເຊິ່ງທັງສອງ ຄ່ານີ້ຈະກ່ຽວຂ້ອງກັບການຫາຄ່າກະຕວງຂອງສົມມຸດຖານ ແລະ ເຫດການທີ່ສົນໃຈແລ້ວ ຈະຊ່ວຍໃຫ້ຮູ້ຄ່າ ຄວາມແນ່ນອນ ດ້ວຍສົມຜົນ

$$CF = \frac{MB,(H,E) - MD(H,E)}{1 - \min[MB(H,E), MD(H,E)]}$$

ໂດຍທີ່ຄ່າ CF ທີ່ໄດ້ຈະນຳໄປໃຊ້ໃນການຫາຄ່າຂອງ CF ທີ່ສົນໃຈພາຍໃຕ້ເງື່ອນໄຂຂອງສົມມຸດຖານໄດ້ ດັ່ງນີ້: **CF(H,E)=CF(E) × CF**

ເຊັ່ນ ຕົວຢ່າງ 4.1 ຄວາມຮູ້ໃນການພະຍາກອນອາກາດອາໄສກົດຕໍ່ໄປນີ້

IF ທ້ອງຟ້າປອດໂປງ

THEN ຈະພະຍາກອນໄດ້ວ່າມື້ນີ້ອາກາດແຈ່ມໃສດີ {CF=0.8 }

ໂດຍທີ່ຄ່າ CF ຂອງທ້ອງຟ້າປອດໂປງ ຄື: 0.5

ດັ່ງນັ້ນ ຈະຫາຄ່າ CF ຂອງອາກາດມື້ນີ້ທີ່ມີເງື່ອນໄຂສາພັນກັບທອງຟ້າປອດໂປງໄດ້ເປັນຕົ້ນ.

CF(H,E)=CF(E) ×CF

CF(H,E)= 0.5×.08

CF(H,E)=0.4

ເມື່ອປຽບທຽບຄ່າ CF ທີ່ໄດ້ຈາກຕາຕະລາງທີ 4.1 ສາມາດສະຫຼຸບໄດ້ວ່າ “ມື້ນີ້ອາກາດອາດຈະແຈ່ມໃສ”.

ດັ່ງທີ່ໄດ້ກ່າວມານັ້ນເປັນພຽງການຫາຄ່າ CF ຈາກຫຼັກການທີ່ບໍ່ມີຄວາມຊັບຊ້ອນປານໃດ, ເຊິ່ງໃນຄວາມ ເປັນຈິງການສ້າງກົດຕ່າງໆ ອາດມີຄວາມຫຍຸ້ງຫຍາກກ່ວານີ້ ການເຊື່ອມຕໍ່ກັບເງື່ອນໄຂອາດມີຫຼາຍກ່ວາ ໜຶ່ງຂໍ້ ເຊັ່ນ: ຖ້າ A ແລະ B ແລ້ວ C ເປັນຕົ້ນ. ໃນກໍລະນີມີຫຼາຍກ່ວາ ໜຶ່ງເຫດການເຊື່ອມຕໍ່ກັບສົມມຸດ ຖານດ່ຽວຫາກຫາຄ່າ CF ຈະແຕກຕ່າງຈາກເດີມໂດຍຈະໃຊ້ສົມຜົນຕໍ່ໄປນີ້:

ຫຼັກການ Conjunction

CF(H,E1 ∧ E2∧ E3,…… ∧ En) =min[CF(E1), CF(E2), ….. CF(En)] × CF

ຫຼັກການ Disjunction

CF(H,E1 ∨ E2∨ E3,…… ∨ En) =max[CF(E1), CF(E2), ….. CF(En)] × CF

ຕົວຢ່າງ 4.2

IF ທ້ອງຟ້າປອດໂປ່ງ

AND ການພະຍາກອນບອກວ່າອາກາດແຈ່ມໃສ

THEN ຄວນຈະໃສ່ແວ່ນຕາກັນແດດ {CF 0.8}

ໂດຍທີ່ຄ່າ CF ຂອງທ້ອງຟ້າປອດໂປ່ງ ຄື: 0.9 ແລະ CF ຂອງການພະຍາກອນວ່າອາກາດແຈ່ມໃສ ຄື: 0.7 ດັ່ງນັ້ນຈະຫາຄ່າ CF ວ່າຄວນຈະໃສ່ແວ່ນກັນແດດພາຍໃຕ້ເຫດການອື່ນໆ.

ໂດຍ E_1 ຄືທ້ອງຟ້າປອດໂປ່ງ ແລະ E_2 ຄື ການພະຍາກອນບອກວ່າອາກາດແຈ່ມໃສ

ຈະໄດ້ວ່າ:

CF(H,E1 ∧ E2) =min[CF(E1), CF(E1)] × CF

=min[0.9,0.7] × 0.8

=0.7x0.8

=0.56

ເມື່ອປຽບທຽບຄ່າ CF ກັບຕາຕະລາງ 4.1 ຈຶ່ງສະຫຼຸບໄດ້ວ່າ “ມື້ນີ້ມີຄວາມເປັນໄປໄດ້ວ່າຄວນຈະໃສ່ແວ່ນກັນແດດ”.

ຈາກທີ່ກ່າວມາຂ້າງເທິງນັ້ນເປັນການຫາຄ່າ CF ເພື່ອພິຈາລະນາສົມມຸດຖານ ແລະ ເຫດການທີ່ສົນໃຈວ່າມີຄວາມເປັນໄດ້ ແລະ ມີຄ່າຄວາມແນ່ນອນທີ່ຈະເກີດຂຶ້ນຫຼາຍນ້ອຍເທົ່າໃດ ເຊິ່ງສາມາດນຳໄປປະຍຸກໃຊ້ກັບຄວາມຮູ້ທີ່ສ້າງຂຶ້ນດ້ວຍຫຼັກການໃນລະບົບຜູ້ຊ່ຽວຊານໄດ້. ຈາກຕົວຢ່າງ ທີ່ຜ່ານມາໄດ້ກຳນົດຄ່າ CF ງ່າຍໆຂຶ້ນມາເພື່ອໃຫ້ເຂົ້າໃຈ ຄວາມຈິງແລ້ວຄ່າກະຕວງຂອງສົມມຸດຖານ ຫຼື ເຫດການທີ່ສົນໃຈສາມາດຄຳນວນຫາໄດ້ໂດຍໃຊ້ທິດສະດີຄ່າກະຕວງຕ່າງໆ. ໃນບົດນີ້ຈະໄດ້ເວົ້າເຖິງເຕັກນິກການຊອກຫາຄ່າກະຕວງພາຍໃຕ້ຫຼັກການຂອງຄວາມບໍ່ແນ່ນອນ ເຊັ່ນ: ທິດສະດີຂອງ Bayes ແລະ ທິດສະດີຂອງ Demster and Shafer ເປັນຕົ້ນ. ທິດສະດີຕ່າງໆເຫຼົ່ານີ້ສະໜັບສະໜູນໃຫ້ການຊອກຫາຄ່າຄວາມແນ່ນອນພາຍໃຕ້ສະຖານະການແວດລ້ອມທີ່ປະກອບດ້ວຍຕົວປ່ຽນທີ່ມີຄວາມບໍ່ແນ່ນອນຢູ່ນຳ ແລະ ສາມາດນຳໄປປະຍຸກໃຊ້ໃນລະບົບຕ່າງໆ ຂອງປັນຍາປະດິດໄດ້ຢ່າງມີປະສິດທິພາບ.

ໃນໂລກຄວາມເປັນຈິງ ຄວາມບໍ່ແນ່ນອນເກີດຂຶ້ນໄດ້ສະເໝີ ເນື່ອງຈາກມະນຸດມີຈິດໃຈ ແລະ ສະໝອງທີ່ສຳພັນກັນຫຍາກທີ່ຈະຕັດຂາດຈາກກັນໄດ້, ສົ່ງຜົນໃຫ້ການຕັດສິນໃຈ ຫຼື ການໃຫ້ຂໍ້ມູນອາດເກີດຄວາມຫຼົງເລ ແລະ ຄວາມບໍ່ແນ່ໃຈຂຶ້ນໄດ້. ບາງຄັ້ງອາດຕັດສິນໃຈດ້ວຍສະໝອງທີ່ຂາດຄວາມຮູ້ ຫຼື ຂາດສະຕິ ແລະ ອາດໃຊ້ອາລົມເໜືອເຫດຜົນ ເຮັດໃຫ້ຂໍ້ມູນບາງປະເພດເກີດຄວາມບໍ່ແນ່ນອນໄດ້ສະເໝີ. ການນຳຂໍ້ມູນທີ່ມີຄວາມບໍ່ແນ່ນອນມາໃຊ້ໃນການຈັດຮູບແບບຄວາມຮູ້ເປັນສ່ວນທີ່ຫຼີກລ້ຽງໄດ້ຫຍາກ ເຮັດໃຫ້ຈຳເປັນຕ້ອງສຶກສາເລື່ອງຄວາມບໍ່ແນ່ນອນໄວ້. ໃນກໍລະນີທີ່ຄວາມບໍ່ແນ່ນອນເປັນຕົວເລກອາດແກ້ໄຂ ຫຼື ຫາທິດສະດີມາສະໜັບສະໜູນໄດ້ ແຕ່ໃນກໍລະນີທີ່ບໍ່ແມ່ນຮູບແບບຂອງຕົວເລກຄ່າຄວາມບໍ່ແນ່ນອນອາດຕ້ອງໃຊ້ຫຼັກການທີ່ແຕກຕ່າງຈາກເດີມ. ດັ່ງນັ້ນຈຳເປັນຕ້ອງນຳປັດໃຈແວດລ້ອມຂອງຄວາມບໍ່ແນ່ນອນມາພິຈາລະນາອີກດ້ວຍເພື່ອໃຫ້ໄດ້ຜົນຮັບທີ່ໃກ້ຄຽງ ແລະ ມີປະສິດທິພາຍຫຼາຍທີ່ສຸດ.

4.2 ທິດສະດີ Bayes

ທິດສະດີ Bayes ເປັນອີກຫຼັກການທີ່ໃຊ້ແກ້ໄຂບັນຫາເລື່ອງຄວາມບໍ່ແນ່ນອນທີ່ອາດເກີດຂຶ້ນຈາກການນຳສະເໜີ ຫຼື ຮຽນຮູ້ຄວາມຮູ້ຕ່າງໆໃນລະບົບປັນຍາປະດິດ. ທິດສະດີ Bayes ເປັນທິດສະດີທາງສະຖິຕິ ໂດຍນຳຄ່າກະຕວງມາໃຊ້ປະເມີນຄວາມບໍ່ແນ່ນອນເປັນຕົວເລກໄດ້.

ທິດສະດີ Bayes ກ່ຽວຂ້ອງໂດຍກົງກັບເງື່ອນໄຂຄວາມທີ່ຄາດວ່າຈະເກີດຂຶ້ນດັ່ງນີ້:

ກຳນົດ A ແລະ B ເປັນເຫດການ ແລະຄ່າກະຕວງທີ່ຈະເກີດເຫດການ A ໂດຍມີເງື່ອນໄຂວ່າເຫດການ B ໄດ້ເກີດຂຶ້ນແລ້ວສາມາດຂຽນແທນດ້ວຍ “P(A/B)” ດັ່ງນັ້ນສາມາດຂຽນທິດສະດີການກະຕວງເປັນສົມຜົນໄດ້ດັ່ງນີ້:

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

ຄ່າກະຕວງຂອງ A ເມື່ອຮູ້ B ຈະຄຳນວນໄດ້ຈາກ ຜົນຄູນຂອງຄ່າກະຕວງຂອງ B ເມື່ອຮູ້ A ກັບຄ່າກະຕວງຂອງ A ທັງໝົດຫານດ້ວຍຄ່າກະຕວງຂອງ B ໃນປັນຍາປະດິດການຮຽນຮູ້ຂອງເຄື່ອງຄອມພິວເຕີຈະກ່ຽວຂ້ອງກັບເຫດການ ແລະ ການຕັ້ງສົມມຸດຖານຕ່າງໆ. ທິດສະດີ Bayes ຈະຊ່ວຍໃນການຊອກຫາສິ່ງທີ່ສົນໃຈຈາກຊຸດຂໍ້ມູນ ຫຼື ຕົວຢ່າງຂໍ້ມູນເພື່ອໃຫ້ໄດ້ຄ່າກະຕວງຂອງສົມມຸດຖານຂອງສິ່ງທີ່ສົນໃຈວ່າມີໂອກາດຈະເກີດຂຶ້ນໄດ້ຫຼາຍນ້ອຍເທົ່າໃດໂດຍສາມາດຄຳນວນຕາມທິດສະດີ Bayes ເຊິ່ງມີສົມຜົນດັ່ງນີ້ [Akerkar, 2005]

$$P(H_i/B) = \frac{P(E/H_i)P(H_i)}{\sum_{n=1}^{k} P(E/H_n)\,P(H_n)}$$

P(H_i) = ຄ່າກະຕວງກ່ອນໜ້າທີ່ສົມມຸດຖານ H ຈະເປັນຈິງໂດຍທີ່ບໍ່ຢູ່ພາຍໃຕ້ເຫດການໃດໆ

P(H_i/E) = ຄ່າກະຕວງກ່ອນໜ້າທີ່ສົມມຸດຖານ H ທີ່ເປັນຈິງເມື່ອຢູ່ພາຍໃຕ້ເຫດການ E

P(E/H_i) = ຄ່າກະຕວງໄດ້ຈາກການສັງເກດເຫດການ E ເຊິ່ງເຮັດໃຫ້ສົມມຸດຖານ H ເປັນຈິງ

k = ຈຳນວນຂໍ້ສົມມຸດຖານທີ່ເປັນໄປໄດ້

ທິດສະດີ Bayes ຈະນຳມາປະຍຸກໃຊ້ກັບປັນຍາປະດິດ ເພື່ອຄຳນວນຫາຕົວເລກຂອງການວິເຄາະ ຫຼື ການບົ່ງມະຕິສົມມຸດຖານຂອງເຫດການຕ່າງໆ ໂດຍ H ຈະເປັນຕົວກຳນົດຜົນຂອງ ເຫດການ E ພາຍໃຕ້ເງື່ອນໄຂທີ່ກຳນົດ ແລະ P(H_i/E) ຈະສະແດງເຖິງຄວາມແນ່ນອນຂອງເຫດການໄດ້ອີກດ້ວຍ. ເຊິ່ງຄວາມສຳພັນດັ່ງກ່າວຊ່ວຍໃນການຕັດສິນໃຈວ່າເຫດການ ຫຼື ຂໍ້ສົມມຸດຖານທີ່ປັນຍາປະດິດພົບພໍ້ມີຄວາມໜ້າເຊື່ອຖືຫຼາຍໜ້ອຍເທົ່າໃດ ເຊັ່ນ ການບົ່ງມະຕິເຊື້ອພະຍາດຂອງແພດເຊິ່ງອາດໃຊ້ເຄື່ອງຄອມພິວເຕີມາຊ່ວຍໃນການການຕັດສິນໃຈວ່າຜູ້ປ່ວຍມີອາການໃກ້ຄຽງກັບພະຍາດ ຫຼື ພະຍາດຊະນິດໃດ

ຕົວຢ່າງ 4.3 ກຳນົດໃຫ້ຊາມສອງໜ່ວຍບັນຈຸໝາກບານເຕັມຊາມ ໂດຍແຕ່ລະຊາມຈະສາມາດບັນຈຸໄດ້ສູງສຸດ 40 ອັນ ແລະ ບັນຈຸໝາກບານ ສອງ ສີ ຄື ສີແດງ ແລະ ສີສົ້ມ, ຊາມແຕ່ລະໜ່ວຍມີໝາຍເລກກຳກັບຢູ່ ຄື ໝາຍເລກ ໜຶ່ງ ແລະ ສອງ ຕາມລຳດັບ. ຕ້ອງການຊອກຫາຄ່າກະຕວງທີ່ຈະຈັບໄດ້ຊາມໝາຍເລກ 1 ແລະ ເປັນໝາກບານ ສີສົ້ມ.

ລາຍການ	ຊາມໝາຍເລກ 1	ຊາມໝາຍເລກ 2	ລວມ
ໝາກບານສີແດງ	10	20	30
ໝາກບານສີສົ້ມ	30	20	50
ລວມ	40	40	80

ກຳນົດໃຫ້ A ໝາຍເລກຊາມ

B ຄື ໝາກບານສີສົ້ມ

ດັ່ງນັ້ນຄ່າກະຕວງທີ່ຈະຈັບໄດ້ຊາມໝາຍເລກ 1 ແລະ ໝາກບານສີສົ້ມ ແມ່ນ

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

ຈາກສູດຂ້າງເທິງຈຳເປັນຕ້ອງຮູ້ຄ່າກະຕວງຕໍ່ໄປນີ້:

1. ຄ່າກະຕວງຂອງການຈັບໄດ້ຊາມໝາຍເລກ 1

ຈຳນວນ ຊາມທັງໝົດມີ 2 ໜ່ວຍ ຄ່າກະຕວງທີ່ຈະຈັບໄດ້ 1 ໜ່ວຍຈາກທັງໝົດ 2 ໜ່ວຍຄື

$$P(A)=1/2 = 0.5$$

2. ຄ່າກະຕວງຂອງການຈັບໄດ້ໝາກບານສີສົ້ມ

ໝາກບານສີສົ້ມມີຢູ່ທັງໃນຊາມໝາຍເລກ 1 ແລະ 2 ດັ່ງນັ້ນ ຄ່າກະຕວງໃນການຈັບໄດ້ໝາກບານສີສົ້ມ ຄື P(B) ໄດ້ຈາກຜົນບວກຂອງຄ່າກະຕວງໃນການຈັບໝາກບານສີສົ້ມຈາກຊາມໝາຍເລກ 1 ແລະ 2 ດັ່ງນີ້:

- ຊາມໝາຍເລກ 1 ໂອກາດທີ່ຈະຈັບໄດ້ ຄື 0.5

ໃນຊາມໝາຍເລກ 1 ມີໝາກບານທັງໝົດ 40 ອັນ ເປັນສີສົ້ມ 30 ອັນ ສະແດງວ່າຄ່າກະຕວງທີ່ຈະຈັບໄດ້ໝາກບານສີສົ້ມ ຄື P(B1)

$$P(B1)=(30/40) \times 0.5 = 0.75\times0.5$$

- ຊາມໝາຍເລກ 2 ໂອກາດທີ່ຈະຈັບໄດ້ ຄື 0.5

  ໃນຊາມໝາຍເລກ 2 ມີໝາກບານທັງໝົດ 40 ອັນ ເປັນສີສົ້ມ 20 ອັນ ສະແດງວ່າຄ່າກະຕວງທີ່ຈະຈັບໄດ້ໝາກບານສີສົ້ມ ຄື P(B2)

$$P(B2)=(20/40) \times 0.5 = 0.5\times0.5$$

ດັ່ງນັ້ນ

$$P(B) = P(B1)+ P(B2)$$

$$=(0.75\times0.5)+(0.5\times0.5)$$

$$=0.375+0.25$$

$$=0.625$$

ຄ່າກະຕວງຂອງການຈັບໄດ້ໝາກບານສີສົ້ມຄື 0.625

3. ຄ່າກະຕວງຂອງການຈັບໄດ້ໝາກບານສີສົ້ມຈາກຊາມໝາຍເລກ 1

ຈາກຂໍ້ 2 ເຮັດໃຫ້ຮູ້ວ່າຄ່າກະຕວງຂອງການຈັບໄດ້ໝາກບານສີສົ້ມຈາກຊາມໝາຍເລກ 1 P(B/A) ຄື 0.375 ຈາກຂໍ້ 1 ເຖິງ 3 ສາມາດຄຳນວນຫາຄ່າກະຕວງທີ່ຕ້ອງການໄດ້ດັ່ງນີ້

$$P(A/B) = \frac{0.375 \times 0.5}{0.625}$$

$$= 0.1875/0.625$$

$$= 0.3$$

ຄ່າກະຕວງຂອງການຈັບໄດ້ຊາມໝາຍເລກ 1 ແລະ ເປັນໝາກບານສີສົ້ມ ແມ່ນ 0.3

ຈາກຕົວຢ່າງ 4.3 ການພິຈາລະນາຄ່າກະຕວງທີ່ສົມໃຈຈຳເປັນຕ້ອງປະກອບດ້ວຍປັດໃຈອ້ອມຂ້າງຫຼາຍຢ່າງ, ເຊິ່ງຕ້ອງອາໄສທິດສະດີຂອງຄ່າກະຕວງໃນການແກ້ໄຂບັນຫາ. ຈາກຕົວຢ່າງເປັນພຽງການປະຍຸກໃຊ້ໃນລະດັບພື້ນຖານເທົ່ານັ້ນນອກຈາກເຕັກນິກດັ່ງກ່າວແລ້ວ ຍັງມີເຕັກນິກອື່ນໆ ທີ່ໃຊ້ຫຼັກການຄ້າຍຄືກັນແມ່ນ **Full Joint Distribution**

Full Joint Distribution ຈາກຕົວຢ່າງຂ້າງເທິງ ຈະເຫັນໄດ້ວ່າທິດສະດີ ສາມາດຫາຄ່າກະຕວງຂອງສົມມຸດຖານພາຍໃຕ້ສະຖານະການຕ່າງໆໄດ້ ເຊິ່ງຕ່າງກໍເປັນເຫດຜົນຊອດຄອງກັນ, ເນື່ອງຈາກການທີ່ເກີດຜົນຮັບໄດ້ ຈຳເປັນຕ້ອງມີສາເຫດທີ່ເປັນຕົ້ນກຳເນີດຂອງເຫດການນັ້ນ. ແຕ່ໃນກໍລະນີຜົນໄດ້ຮັບເກີດຈາກສາເຫດ ຫຼື ເຫດການຫຼາຍກ່ວາໜຶ່ງໃນເວລາດ່ຽວກັນ ຈຳເປັນຕ້ອງລວບລວມເຫດການທີ່ໃກ້ຄຽງກັນເພື່ອໃຫ້ສາມາດຊອກຫາຄ່າກະຕວງຂອງສົມມຸດຖານໄດ້ຢ່າງຖືກຕ້ອງ. ໃນການລວມເຫດການອາດຕ້ອງອາໄສວິທີການແຍກບາງຄ່າກະຕວງຂອງແຕ່ລະເຫດການເຊິ່ງຊ່ວຍໃຫ້ເຂົ້າໃຈ ແລະ ຄຳນວນໄດ້ງ່າຍຂຶ້ນ ວິທີການນີ້ເອີ້ນວ່າ Full Joint Distribution ເປັນວິທີການແຈກຢາຍຄ່າກະຕວງທີ່ໄດ້ຈາກການສັງເກດ ຫຼື ການຈຳລອງທີ່ນຳໄປໃຊ້ໃນການຄຳນວນຕໍ່ໄປ.

ການແຈກຢາຍຂໍ້ມູນຈະບັນທຶກລົງໃນຕາຕະລາງຕາມຕົວປ່ຽນທີ່ມີ ຫຼື ທີ່ໃຫ້ຄວາມສົນໃຈດັ່ງຕາຕະລາງລຸ່ມນີ້:

ອາການທີ່ກວດພົບ	ເຈັບແຂ້ວ		ບໍ່ເຈັບແຂ້ວ	
	ມີຫີນປູນ	ບໍ່ມີຫີນປູນ	ມີຫີນປູນ	ບໍ່ມີຫີນປູນ
ເຫືອກອັກເສບ	0.108	0.012	0.072	0.008
ເຫືອກບໍ່ອັກເສບ	0.016	0.064	0.144	0.576

ຕາຕະລາງ 4.3 ສະແດງຄ່າແຈກຢາຍຄ່າກະຕວງຂອງອາການພາຍໃນຊ່ອງປາກ [Russell and Norving, 2003]

ຈາກຕາຕະລາງຄ່າກະຕວງຂອງແຕ່ລະເຫດການຈະຖືກແຈກຢາຍໄວ້ຢ່າງຈະແຈ້ງເມື່ອຕ້ອງການພິຈາລະນາເຫດການທີ່ມີຄວາມສຳພັນກັນຈະເບິ່ງໄດ້ຈາກ ແຖວ ແລະ ຖັນ ເຊິ່ງຈະຊ່ວຍໃຫ້ສາມາດຄຳນວນຫາຄ່າກະຕວງຂອງຂໍ້ສົມມຸດຖານທີ່ຢູ່ພາຍໃຕ້ເຫດການຫຼາຍກ່ວາໜຶ່ງຢ່າງໄດ້ດັ່ງຕົວຢ່າງ 4.4

ຕົວຢ່າງ 4.4 ພິຈາລະນາຄ່າກະຕວງຂອງອາການຄົນເຈັບທີ່ພົບທັນຕະແພດດັ່ງຕາຕະລາງທີ່ 4.3

ຈົ່ງຫາຄ່າກະຕວງຂອງຄົນເຈັບທີ່ມີອາການເຫືອກອັກເສບ ຫຼືເຈັບ ແຂ້ວ

ຄ່າກະຕວງຂອງຄົນເຈັບທີ່ມີອາການເຫືອກອັກເສບ ຫຼື ເຈັບແຂ້ວຊອກໄດ້ຈາກຜົນບວກຂອງຄ່າກະຕວງທັງໝົດຂອງຄົນເຈັບທີ່ມີອາການເຫືອກອັກເສບ ແລະ ຜົນບວກຂອງຄ່າກະຕວງຂອງຄົນເຈັບທີ່ອາການເຈັບແຂ້ວ ເນື່ອງຈາກເປັນເຫດການທັງໝົດທີ່ສົນໃຈ, ເຊິ່ງພິຈາລະນາຄ່າກະຕວງທີ່ແຈກຢາຍໃນຕາຕະລາງຕາມທີ່ສິ່ງທີ່ສົນໃຈເຮັດໃຫ້ໄດ້ຜົນຮັບດັ່ງນີ້:

ກຸ່ມເຫືອກອັກເສບ ເຈັບແຂ້ວ ແລະມີຫີນປູນ (0.108)

ກຸ່ມເຫືອກອັກເສບ ເຈັບແຂ້ວ ແຕ່ບໍ່ມີຫີນປູນ (0.012)

ກຸ່ມເຫືອກອັກເສບ ມີຫີນປູນ ແຕ່ບໍ່ເຈັບແຂ້ວ (0.072)

ກຸ່ມເຫືອກອັກເສບ ພຽງຢ່າງດ່ຽວ (0.008)

ໃນຂະນະທີ່ກຸ່ມຄົນເຈັບທີ່ເຈັບແຂ້ວ (ໂດຍບໍ່ສົນໃຈວ່າຈະມີອາການອື່ນຫຼືບໍ່) ມີທັງໝົດ 4 ກຸ່ມເຊັ່ນດຽວກັບຄົນເຈັບເຫືອກອັກເສບ ແຕ່ໃນກຸ່ມຄົນເຈັບທີ່ເຈັບແຂ້ວຈະມີ 2 ກຸ່ມທີ່ມີອາການເຫືອກອັກເສບ ເຊິ່ງຊ້ຳກັບກຸ່ມທີ່ກ່າວໄປແລ້ວໃນເບື້ອງຕົ້ນ.

ດັ່ງນັ້ນ ກຸ່ມຂອງອາການເຈັບແຂ້ວຈຶ່ງເຫຼືອພຽງ 2 ກຸ່ມຄື: ກຸ່ມເຈັບແຂ້ວມີຫີນປູນ ແຕ່ເຫືອກບໍ່ອັກເສບ (0.016) ແລະ ກຸ່ມທີ່ເຈັບແຂ້ວພຽງຢ່າງດຽວ (0.064).

ຈຶ່ງສະຫຼຸບໄດ້ວ່າ ຄົນເຈັບທີ່ມີເຫືອກອັກເສບ ຫຼື ເຈັບແຂ້ວ ໂດຍບໍ່ສົນໃຈອາການອື່ນໆ ມີທັງໝົດ 6 ກຸ່ມດັ່ງນັ້ນ ຄ່າກະຕວງຂອງຄົນເຈັບທີ່ມີເຫືອກອັກເສບ ຫຼື ເຈັບແຂ້ວ

=0.108+0.012+0.072+0.008+0.016+0.064

= 0.28

ຈາກຕົວຢ່າງທີ່ໄດ້ກ່າວມາຈະເຫັນໄດ້ວ່າການລວມເຫດການແບບ ຈະຊ່ວຍໃຫ້ການຄຳນວນເຮັດໄດ້ຢ່າງວ່ອງໄວຂຶ້ນເພາະໂອກາດທີ່ສົມມຸດຖານຈະເກີດຂຶ້ນພາຍໃຕ້ເຫດການຫຼາຍກ່ວາໜຶ່ງຢ່າງນັ້ນເປັນໄປໄດ້ງ່າຍໂດຍສະເພາະການປະຍຸກໃຊ້ວຽກດ້ານລະບົບປັນຍາປະດິດ.

4.3 ທິດສະດີ Dempster and Shafer

ທິດສະດີ Dempster and Shafer ເອີ້ນສັ້ນໆວ່າ " ທິດສະດີ D-S" ເປັນທິດສະດີທາງຄະນິດສາດທີ່ຖືກພັດທະນາຂຶ້ນມາເພື່ອໃຊ້ໃນການອະທິບາຍຄ່າຄວາມບໍ່ແນ່ນອນ ເຊິ່ງທຳອິດຖືກພັດທະນາໂດຍ Arthur Dempster ແລະຖືກນຳມາພັດທະນາຕໍ່ໂດຍ Glenn Shafer ເຊິ່ງໄດ້ນຳສະເໜີການອະທິບາຍຄ່າຄວາມບໍ່ແນ່ນອນດ້ວຍຟັງຊັນພິເສດທີ່ເອີ້ນວ່າ "Belief Function". ທິດສະດີ D-S ກ່ຽວຂ້ອງກັບການແກ້ໄຂບັນຫາເລື່ອງຄ່າຄວາມບໍ່ແນ່ນອນທີ່ເກີດຂຶ້ນໃນການຕັ້ງສົມມຸດຖານ ໂດຍທິດສະດີ D-S ນີ້ຈະກຳນົດຊ່ວງຕົວເລກຂອງຄ່າກະຕວງເພື່ອໃຊ້ໃນການສະໜັບສະໜູນສົມມຸດຖານ.

ໃນທິດສະດີ D-S ຈະມີຄວາມກ່ຽວຂ້ອງກັບກຸ່ມຂອງເຫດການ (Evidence) ຈຶ່ງຈຳເປັນຕ້ອງມີຄວາມຮູ້ພື້ນຖານໃນເລື່ອງກຸ່ມ (Set) ດ້ວຍໂດຍແທນເຫດການໜຶ່ງໆ ເປັນສະມາຊິກພາຍໃນກຸ່ມ, ເຊິ່ງຈະຊ່ວຍໃຫ້ການພິຈາລະນາເຫດການທັງໝົດທີ່ສາມາດເກີດຂຶ້ນໄດ້ງ່າຍ, ຄວາມຮູ້ເບື້ອງຕົ້ນໃນທິດສະດີກຸ່ມທີ່ຈຳເປັນຕ້ອງສຶກສາມີດັ່ງນີ້:

ເມື່ອ $A=\{x,y,z\}$

ແລ້ວ Power set ຈະເປັນດັ່ງນີ້ $P(A)=\{\phi,\{x\},\{y\},\{z\},\{x,y\},\{x,z\},\{y,z\},\{x,y,z\}\}$

ການພິຈາລະນາໃນທິດສະດີ D-S ຈະດຳເນີນການທີ່ສະມາຊິກພາຍໃນກຸ່ມ, ເຊິ່ງສະມາຊິກດັ່ງກ່າວອາດໝາຍເຖິງ ຄົນ,ສັດ, ສິ່ງຂອງ ເຫດການຕ່າງໆໂດຍກຸ່ມທີ່ບັນຈຸສິ່ງເຫຼົ່ານີ້ໃນທິດສະດີ D-S ເອີ້ນວ່າ "Environment" ເຊິ່ງຈະຂຽນແທນດ້ວຍເຄື່ອງໝາຍ Θ ດັ່ງນີ້:

$$\Theta = \{ \theta_1, \theta_2, \theta_3, \ldots\ldots\ldots\ldots\ldots, \theta_n \}$$

$\theta_1, \theta_2, \theta_3, \ldots\ldots\ldots\ldots\ldots, \theta_n$ ໝາຍເຖິງ ສະມາຊິກຂອງກຸ່ມທີ່ສາມາດແທນຄວາມໝາຍໄດ້ຫຼາຍຮູບແບບຂຶ້ນຢູ່ກັບເຫດການ ຫຼື ສິ່ງຂອງທີ່ເຮົາສົນໃຈດັ່ງຕົວຢ່າງທີ່ 4.5

ຕົວຢ່າງ 4.5

Θ = {Car,Van,Bike}

ແທນຄ່າດັ່ງນີ້

ລົດ = C(Car) , ລົດຕູ້ = V(Van) , ແລະ ລົດຈັກ = B(Bike)

ຫາກສະຖານະການທີ່ເຮົາສົນໃຈເປັນດັ່ງນີ້:

ພາຫານະໃດທີ່ມີຫຼາຍກ່ວາ 2 ລໍ້ ຄຳຕອບຄື $\{ \theta_1, \theta_2 \} = \{C,V\}$

ພາຫານະໃດທີ່ບັນທຸກຄົນຫຼາຍກ່ວາ 2 ຄົນ ແລ້ວຜິດກົດໝາຍ ຄຳຕອບຄື $\{ \theta_3 \} = \{B\}$

Environment ດັ່ງກ່າວ ສາມາດຂຽນກຸ່ມສະແດງເຖິງສະມາຊິກທັງໝົດໄດ້ດັ່ງນີ້:

P(Θ) = $\{\phi, \{C\}, \{V\}, \{B\}, \{C\}, \{C,V\}, \{C,B\}, \{V,B\}, \{C,V,B\}\}$

ເຫດການທີ່ສົນໃຈຈະເປັນເງື່ອນໄຂໃນການຈຳແນກສະມາຊິກທີ່ມີຄວາມກ່ຽວຂ້ອງພາຍໃນກຸ່ມນັ້ນອອກມາ ເຊິ່ງຈະເຫຼືອພຽງແຕ່ສະມາຊິກທີ່ມີຄ່າກະຕວງທີ່ເຮົາສົນໃຈ. ດັ່ງນັ້ນຈຳນວນສະມາຊິກພາຍໃນກຸ່ມປ່ຽນແປງໄປຕາມຄວາມຕ້ອງການ ຫຼື ຂໍ້ຈຳກັດຂອງເຫດການທີ່ສົນໃຈ. ຢ່າງໃດກໍຕາມສະມາຊິກທີ່ຖືກຈຳແນກອອກມານັ້ນຖືເປັນກຸ່ມຍ່ອຍ (Subset) ຂອງ Θ ນັ້ນເອງ.

ທິດສະດີທີ່ສຳຄັນອັນຂອງ D-S ທີ່ຈຳເປັນຕ້ອງສຶກສາ ແລະ ທຳຄວາມເຂົ້າໃຈຄື: ຮູບແບບຂອງຟັງຊັນຕ່າງໆທີ່ມີຄວາມສຳຄັນຕໍ່ການລະບຸຄ່າຂອງເຫດການໂດຍນຳຄ່າກະຕວງມາພິຈາລະນາ ແລະ ຄຳນວນຫາຕາມຮູບແບບຂອງແຕ່ລະຟັງຊັນ, ສຳລັບຟັງຊັນທີ່ໄດ້ໃນ D-S ມີດັ່ງນີ້:

4.3.1. Mass Function

ໃນທິດສະດີຂອງ D-S ຈະກ່ຽວຂ້ອງໂດຍກົງກັບຄ່າຄວາມເຊື່ອ ຫຼື ລະດັບຄວາມເຊື່ອ (Degree of Belief) ເຊິ່ງຈະເປັນຕົວບົ່ງບອກຄວາມເປັນໄປໄດ້ຂອງເຫດການທີ່ສາມາດເກີດຂຶ້ນໄດ້ ແຕ່ຄ່າຄວາມເຊື່ອນັ້ນມີຄວາມຄາດເຄື່ອນສູງ ຈຶ່ງຈຳເປັນຕ້ອງມີຟັງຊັນທີ່ມາສະໜັບສະໜູນເຫດການນັ້ນ Mass Function

ເປັນຟັງຊັນທີ່ຄ້າຍກັບການບົ່ງບອກເຖິງປະລິມານຂອງມວນສານຂອງວັດຖຸໃນທາງກາຍະພາບ. ສຳລັບ D-S ແລ້ວຄ່າສ່ວນຫຼາຍມັກຈະກ່ຽວຂ້ອງກັບຄ່າກະຕວງ ເຊິ່ງມີຄວາມຊັບຊ້ອນຫຼາຍໃນການຄິດຄຳນວນ ດັ່ງນັ້ນຈຶ່ງໃຊ້ ເປັນຕົວແທນເພື່ອໃຫ້ພິຈາລະນາຄ່າໄດ້ຢ່າງຈະແຈ້ງ, ຊ່ວຍໃຫ້ການລວມ (Combine) ແລະ ແຍກ (Split) ຄ່າໃນເຫດການເຫຼົ່ານັ້ນງ່າຍຂຶ້ນ, ຄືກັນກັບການແບ່ງວັດຖຸໜຶ່ງທີ່ມີຄວາມຊັບຊ້ອນ ເພື່ອໃຫ້ງ່າຍຕໍ່ການວັດແທກຄ່າປະລິມານທີ່ແນ່ນອນຂອງວັດຖຸນັ້ນເອງ Mass Function ເອີ້ນສັ້ນໆ ວ່າ “m-function” .

ສຳລັບ D-S ເຖິງແມ່ນຈະກ່ຽວຂ້ອງກັບຄ່າຄວາມເຊື່ອຖື (Belief) ແຕ່ດ້ວຍທິດສະດີຂອງ D-S ແລ້ວບໍ່ສາມາດບັງຄັບໃຫ້ເກີດການຍອມຮັບຫຼືປະຕິເສດສົມມຸດຖານໄດ້ໂດຍກົງ, ສິ່ງທີ່ຕ້ອງການນຳສະເໜີຄື ການວັດແທກປະລິມານຄ່າຂອງຄວາມເຊື່ອຖືພາຍໃຕ້ສະພາບແວດລ້ອມທີ່ຖືກກຳນົດຂຶ້ນໃຫ້ຜູ້ສົນໃຈຕັດສິນ ວ່າ ເຊື່ອຖື (Belief) ຫຼື ບໍ່ເຊື່ອຖື (Nonbelief) , ເຊິ່ງ ໃນທີ່ນີ້ຈະໝາຍເຖິງຄ່າຂອງຄວາມເຊື່ອຕົວໃດທີ່ບໍ່ເປັນກຸ່ມຍ່ອຍພາຍໃຕ້ກຸ່ມຂອງ ແວດລ້ອມ ທີ່ພິຈາລະນາ ຫຼື ສົນໃຈ. ໃນກໍລະນີທີ່ຄວາມເຊື່ອເກີດຂັດແຍ່ງ ຫຼື ປະຕິເສດສົມມຸດຖານຈະເວົ້າໄດ້ວ່າສົມມຸດຖານນັ້ນເປັນຄ່າ ຍັງບໍ່ເຊື່ອຖືໄດ້ (Disbelief) ເຊິ່ງບໍ່ໄດ້ໝາຍ ຄວາມວ່າເປັນຄ່າ ບໍ່ເຊື່ອຖືເລີຍ (Nonbelief) ຫຼື ສົມມຸດຖານນັ້ນບໍ່ເປັນຈິງ. ຄ່າຂອງຄວາມເຊື່ອຖື (Belief) ແລະ ບໍ່ເຊື່ອຖືເລີຍ (Nonbelief) ສະແດງດັ່ງຕົວຢ່າງທີ່ 4.6

ຕົວຢ່າງທີ່ 4.6

ຈາກແວດລ້ອມໃນຕົວຢ່າງທີ່ 4.5

$$\Theta = \{Car,Van,Bike\}$$

ແທນຄ່າດັ່ງນີ້

ລົດ = C(Car) , ລົດຕູ້ = V(Van) , ແລະ ລົດຈັກ = B(Bike)

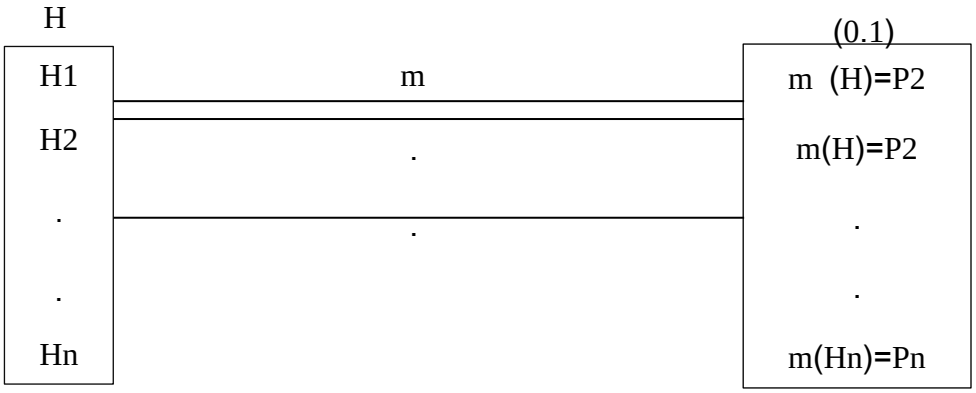
ຫາກສົມມຸດໃຫ້ຄອບຄົວໜຶ່ງມີພາຫະນະທັງໝົດ 3 ຢ່າງຄື ລົດ, ລົດຕູ້, ແລະ ລົດຈັກ ໂດຍມີຄ່າກະຕວງ ທີ່ສະມາຊິກໃນເຮືອນທັງ 3 ຄົນຈະເດີນທາງດ້ວຍພາຫະນະດ່ຽວກັນ ຄື 0.7 ເມື່ອພິຈາລະນາຈາກ ພາຫະນະໃນແວດລ້ອມ (Environment) ແລ້ວຈະເຮັດໃຫ້ຮູ້ວ່າພາຫະນະທີ່ສາມາດບັນທຸກສະມາຊິກໄປ ພ້ອມກັນໄດ້ທັງໝົດ ຄື ລົດ, ແລະ ລົດຕູ້ ໝາຍເຖິງ *{C,V}* ເຊິ່ງຂຽນເປັນ Mass Function ໄດ້ວ່າ:

ຈາກຄ່າດັ່ງກ່າວເຮັດໃຫ້ສາມາດຫາຄ່າ Nonbelief ໄດ້ຈາກຄ່າຂອງ Environment ທັງໝົດຄື 1 ຕາມທິດສະດີຄວາມໜ້າຈະເປັນ P(H)+P(H)=1 ເມື່ອແທນຄ່າແລ້ວຈະໄດ້ດັ່ງນີ້

$$m_1(\Theta)=1-0.7=0.3$$

ຈາກຕົວຢ່າງທີ່ກ່າວມາທາງທິດສະດີໜ້າຈະເປັນຄ່າ 0.7 ຈະໝາຍເຖິງຄວາມເປັນໄປໄດ້ ຫຼື ຄວາມເຊື່ອຂອງເຫດການນັ້ນທີ່ສາມາດເກີດຂຶ້ນໄດ້ ແລະ ຄ່າ 0.3 ຈະໝາຍເຖິງຄວາມບໍ່ເຊື່ອຖືຫຼືໂອກາດທີ່ເຫດການນັ້ນຈະບໍ່ເກີດຂຶ້ນແຕ່ໃນທິດສະດີ D-S ແມ່ນ 0.7 ຄື Belief ແລະ 0.3 ຄື Nonbelief.

ທິດສະດີ D-S ຈະອາໄສທິດສະດີພື້ນຖານຂອງຄວາມໜ້າຈະເປັນໃນການອະທິບາຍຄ່າຂອງເຫດການທີ່ໜ້າສົນໃຈໄດ້ ໂດຍຈະໃຊ້ຄວບຄູ່ກັບ Belief Function ເຊິ່ງຟັງຊັນຄວາມໜ້າຈະເປັນທີ່ຖືກນໍາມາໃຊ້ຄື m-function ເຊິ່ງໝາຍເຖິງຟັງຊັນຂອງຄວາມໜ້າຈະເປັນຂອງສົມມຸດຖານທີ່ສາມາດອະທິບາຍໄດ້ດ້ວຍຮູບຕໍ່ໄປນີ້:



ຮູບທີ 4.1 ສະແດງກຣາບຄຸນສົມບັດຂອງ m-function

ຄຸນສົມບັດຂອງ m-function

1. $0 \leq m(P) \leq 1$
2. $M(\emptyset)=0$
3. $\sum_{P \in H} m(p) = 1$

ໂດຍທີ່ H ໝາຍເຖິງສົມມຸດຖານ

P ໝາຍເຖິງເຫດການທັ້ງໝົດທີ່ສົນໃຈ ຫຼື ກ່ຽວຂ້ອງກັບສົມມຸດຖານ

ຈາກທີ່ໄດ້ກ່າວໄວ້ໃນເບື້ອງຕົ້ນວ່າທິດສະດີ D-S ກ່ຽວຂ້ອງກັບຟັງຊັນຊະນິດໜຶ່ງທີ່ເອີ້ນວ່າ Belief Function ເຊິ່ງມີຄວາມສໍາພັນກັບ m-function ດັ່ງນັ້ນໃນການສຶກສາທິດສະດີ D-S ຈໍາເປັນຕ້ອງເຂົ້າເຖິງນິຍາມຂອງ Belief Function And plausibility Function ດ້ວຍ.

4.3.2 Belief Function

ຄ່າຂອງ Belief Function ໄດ້ຈາກການຮວມກັນຂອງ m-function ຂອງເຫດການຍ່ອຍທັງໝົດທີ່ໜ້າສົນໃຈຫຼືກ່ຽວຂ້ອງກັບສົມມຸດຖານໂດຍ Belief Function ແຂນແທນດ້ວຍ ‘bel(A)’ ຈາກທີ່ກ່າວມາຫາກໍານົດ A ເປັນເຫດການສົມມຸດຖານໃດໆ ແລະ B ເປັນເຫດການຍ່ອຍໃນ A ແລ້ວຈະໄດ້ສົມຜົນດັ່ງນີ້

$$bel(A) = \sum_{B \subseteq A} m(B)$$

4.3.3 Plausibility Function

Plausibility Function ນີ້ຈະໄດ້ຈາກຜົນລວມຂອງເຫດການຍ່ອຍທີ່ສົນໃຈເຊິ່ງມີຄວາມສຳພັນກັບສົມມຸດຖານໂດຍທີ່ວ່າຄືເປັນຄ່າຂອງຜົນລວມຈາກການ Intersect ກັນລະຫວ່າງເຫດການຍ່ອຍກັບສົມມຸດຖານ Plausibility function ຂຽນແທນດ້ວຍ 'pl(A)' ເຊິ່ງຈະຂຽນເປັນສົມຜົນໄດ້ດັ່ງນີ້:

$$pl(A) = \sum_{B \cap A \neq \emptyset} m(B)$$

ສົມຜົນຄວາມສຳພັນລະຫວ່າງ Belief function ແລະ Plausibility function ມີດັ່ງນີ້

$$Pl(A) = 1\text{-}bel(A)$$

ໂດຍທີ່ $\bar{A}$ ຄື Compliment ຂອງ A

- **Combining Evidence**

ເປັນທິດສະດີລວມເຫດການຂອງ D-S ເຊິ່ງຈະນຳເຫດການທີ່ສົນໃຈທັ້ງໝົດມາລວມເຂົ້າກັນດ້ວຍເພື່ອໃຫ້ສາມາດປະເມີນຄ່າຂອງຄວາມເຊື່ອໃນເຫດການເຫຼົ່ານັ້ນໄດ້ງ່າຍຂຶ້ນ ໂດຍໃຊ້ກົດທີ່ເອີ້ນວ່າ "Dumpster's Rule of Combination" ເປັນການລວມ Mass function ເຂົ້າດ້ວຍກັນດັ່ງນີ້:

$$m_1 \oplus m_2(Z) = \sum_{X \cap Y = Z} m_1(X) m_2(Y)$$

ຈາກກົດດັ່ງກ່າວຈະເຫັນໄດ້ວ່າການຮວມລວມເຫດການໄດ້ມາຈາກຜົນຂອງຄ່າຕ່າງໆທີ່ກ່ຽວຂ້ອງກັບເຫດການທີ່ສົນໃຈທັງໝົດພາຍໃຕ້ເງື່ອນໄຂ $X \cap Y = Z$ ສຳລັບເຄື່ອງໝາຍ $\oplus$ ນີ້ເອີ້ນວ່າ "Direct Sum" ເຊິ່ງເປັນເຄື່ອງໝາຍແທນການຮວມຂອງເຫດການຕາມສົມຜົນດ້ານຂວາ ໂດຍຄ່າທີ່ໄດ້ເປັນຄ່າທີ່ເກີດຈາກປະລິມານຂອງເຫດການສ່ວນຫຼາຍທີ່ສອດຄ້ອງກັບຄວາມສົນໃຈທີ່ກຳນົດຂຶ້ນເນື່ອງຈາກເປັນການຮວມດ້ວຍການ Intersection ($\cap$)ຂອງສອງເຫດການດັ່ງຕົວຢ່າງທີ 4.7

ຕົວຢ່າງທີ 4.7 ຈາກ Environment ໃນຕົວຢ່າງທີ 4.5

$$\Theta = \{Car, Van, Bike\}$$

ແທນຄ່າດັ່ງນີ້

ລົດ =C(Car), ລົດຕູ້ = V (Van) ແລະ ລົດຈັກ = B (Bike)

ສົມມຸດໃຫ້ຄອບຄົວໜຶ່ງມີພາຫະນະ 3 ຢ່າງຄື ລົດ, ລົດຕູ້ ແລະ ລົດຈັກໂດຍມີຄວາມໜ້າຈະເປັນສະມາຊິກໃນບ້ານທັງ 3 ຄົນຈະເດີນທາງດ້ວຍລົດຄື 0.9ເຊິ່ງກຳນົດໃຫ້ເຫດການນີ້ເປັນ m_2 ແລະ ຕ້ອງການລວມເຫດການ m_1 (ຈາກຕົວຢ່າງທີ 6,7) ແລະ m_2 ດັ່ງນີ້:

$m_1(\{C,V\})$ $m_1(\Theta)=0,3$

ແລະ m_1 ຄື $m_2(\Theta)=1-0,9=0.1$

$M_2(\{C\})=0.9$ $m_2(\Theta)=0.1$

ຈາກຄ່າເຫດການທັງ m_1 ແລະ m_2 ສາມາດນຳມາຂຽນເປັນຕາຕະລາງເພື່ອສະແດງຄ່າປະເມີນຂອງເຫດການດ້ວຍການ Intersection ດັ່ງນີ້:

Intersection	$m_2(\{c\})=0.9$	$m_2(\Theta)=0.1$
$m_1(\{c,v\}) = 0.7$	$\{c\}= 0.63$	$\{c,v\}=0.07$
$m_1(\Theta)=0,3$	$\{c\}= 0.27$	$(\Theta)=0,3$

ຕາຕະລາງທີ 4.4 ສະແດງການ Intersection ຂອງ m_1 ແລະ m_2

ຄ່າພາຍໃນຕາຕະລາງມາຈາກສົມການ Dumpster Combination ດັ່ງນີ້

$T_{11}(\{C\}) = m_1 (\{C,V\})\ m_2(C) = (0.7)(0.9)=0.63$

$T_{21}(\{B\}) = m_1(\Theta)\ m_2(C) = (0.3)(0.9) = 0.27$

$T_{12}(\{C,V\}) = m_1(\{C,V\})\ m_2(\Theta) = (0.7)(0.1) = 0.07$

$T_{22}(\{\Theta\}) = m_1(\Theta)\ m_2(\Theta) = (0.3)(0.1) = 0.03$

ໂດຍT_{ij} ແທນທີ່ໄດ້ຈາກການ Intersection ໄດ້ຈາກຕາຕະລາງທີ 4.4 ເຊິ່ງ i ຄືແຖວ ແລະ j ຄື ຄໍລຳ

ຈາກຕາຕະລາງທີ 4.4 ຈະເຮັດໃຫ້ຮູ້ຄ່າຂອງ Mass Function .ໃໝ່ຄື: m_3 ພາຍໃຕ້ເຫດການທີ່ແຕກຕ່າງກັນດັ່ງນີ້

$m_3(\{C\}) = m_1 \oplus m_2(C) = 0.63+0.27 = 0.27$

$m_3(\{C,V\}) = m_1 \oplus m_2(C,V) = 0.07$

$m_3(\Theta) = m_1 \oplus m_2(\Theta) = 0.03$

ຈາກທີ່ກ່າວມາຂ້າງເທິງການລວມເຫດການຈະຊ່ວຍສະໜັບສະໜູນໃນການຄຳນວນຄ່າຂອງ Belief function ແລະ plausibility function ໃຫ້ງ່າຍຂຶ້ນໂດຍຄ່າຂອງທັງສອງຟັງຊັນຈະບົ່ງບອກເຖິງຜົນໃນການປະເມີນຄວາມເຊື່ອຂອງເຫດການເຫຼົ່ານັ້ນໄດ້ ເຊິ່ງຈະພິລາລະນາຈາກຄ່າໄດ້ດັ່ງນີ້:

$0<bel(A) < 1$ ໝາຍຄວາມວ່າສະໜັບສະໜູນຄ່າຄວາມເຊື່ອ

$0<pl(A) < 1$ ໝາຍຄວາມວ່າປະຕິເສດຄ່າຄວາມເຊື່ອ

$0<bel(A) \le pl(A) < 1$ ໝາຍຄວາມວ່າທັງສະໜັບສະໜູນ ແລະ ປະຕິເສດຄ່າຄວາມເຊື່ອ

ຈາກຫຼັກການທີ່ກ່າວມາຂອງປະຕິທິນ D-S ຈະເຫັນວ່າມີຄວາມກ່ຽວຂ້ອງໂດຍຕົງກັບພື້ນຖານຂອງຄວາມໜ້າຈະເປັນ ເພື່ອຫາກ່ວ່າຄ່າຄວາມເຊື່ອຂອງເຫດການຢູ່ໃນລະດັບທີ່ສາມາດສະໜັບສະໜູນຂໍ້ມູນຫຼືຄວາມຮູ້ດັ່ງກ່າວໄດ້ຫຼືບໍ ເຊິ່ງຈະເຮັດໃຫ້ຮູ້ວ່າມີຄວາມເປັນໄປໄດ້ຫຼາຍໜ່ອຍຊ່ຳໃດທີ່ຈະເກີດຂຶ້ນ

4.4 Bayesian Network

Bayesian Network ເປັນແບບຂອງຈຳລອງ ກຣາບຂອງຄວາມໜ່ສຈະເປັນ (Probabilistic Graphical Model) ຫຼື ເອີ້ນອີກຊື່ໜຶ່ງວ່າ “Belief Network” ເຊິ່ງເປັນວິທີການນຳສະເໜີກຸ່ມຂອງຕົວປ່ຽນ ແລະ ຄ່າຄວາມໜ້າຈະເປັນທີ່ອິດສະຫຼະຕໍ່ກັນ Bayesian Network ຈະນຳຄວາມຮູ້ທົ່ວໄປທີ່ມີຄວາມສຳພັນ ຫຼື ກ່ຽວຂ້ອງກັນມານຳສະເໜີເປັນເຄືອຂ່າຍທີ່ເຊື່ອມຕໍ່ກັນດ້ວຍຫຼັກຄວາມໜ້າຈະເປັນ ແລະ ຂອງຫຼັກຂອງເຫດຜົນໂດຍອາໄສຫຼັກການດັ່ງນີ້ [Kerkira, 2005]

- ຄວາມຮູ້ຕ່າງໆ ໃນໂລກນີ້ລ້ວນປະກອບຂຶ້ນຈາກໜ່ວຍຂອງເຫດການຫຼືໜ່ວຍຂອງອົງປະກອບຕ່າງໆ
- ແບບຈຳລອງຈະນຳສະເໜີເຫດການຕ່າງໆ ທີ່ກ່ຽວຂ້ອງແລະສົ່ງຜົນກະທົບຕໍ່ກັນ
- ເຫດການໜຶ່ງ ອາດເປັນອິດສະຫຼະກັບເຫດການອື່ນ ແຕ່ອາດມີຄວາມສຳພັນກັບບາງເຫດການໄດ້
- ບາງເຫດການອາດມີທິດທາງຄວາມສຳພັນກັບເຫດການອື່ນພຽງທາງດຽວ ໃນຂະນະທີ່ບາງເຫດການອາດມີຄວາມສຳພັນຫຼາຍກວ່າໜຶ່ງທິດທາງ
- ແຕ່ລະເຫດການສາມາດເຊື່ອມຕໍ່ກັນເປັນແບບຈຳລອງລັກສະນະເຄືອຂ່າຍໄດ້

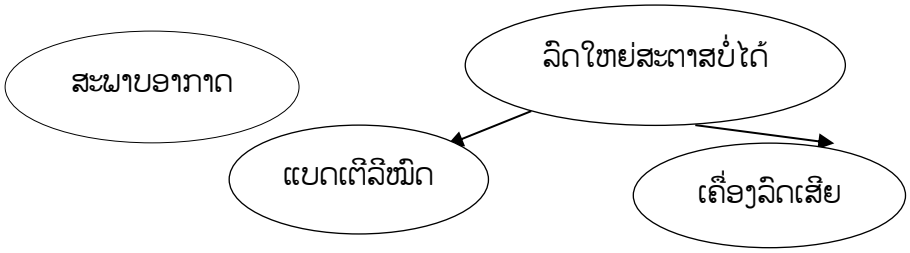
Bayesian Network ເປັນແບບຈຳລອງເຄືອຂ່າຍທີ່ມີຮູບແບບບໍ່ເປັນວົງວຽນ (Cycle) ກ່າວຄືຄວາມສຳພັນຂອງແຕ່ລະໂນດຈະບໍ່ວົນກັບໄປຫາໂນດເດີມໂດຍແຕ່ລະໂນດຈະມີຄວາມສຳພັນກັນຕາມທິດທາງທີ່ແບບຈຳລອງນຳສະເໜີເຊິ່ງບາງໂນດອາດມີຄວາມເປັນອິດສະຫຼະຈາກໂນດອື່ນໆໂດຍທີ່ໂນດທັງໝົດໃນແບບຈຳລອງສຳພັນຫຼືບໍ່ສຳພັນທັງໝົດກໍ່ໄດ້ການສຶກສາເຖິງຫຼັກການຂອງ Bayesian Network ໂດຍມີຄຸນສົມບັດທີ່ສຳຄັນດັ່ງນີ້

4.4.1 ໂຄງສ້າງຂອງ Bayesian Network

ຈາກທີ່ກ່າວມາໃນຫົວຂໍ້ 4.2 ຈະເຫັນໄດ້ວ່າທິດສະດີຂອງ Bayes ຈະມີຄວາມກ່ຽວຂ້ອງກັບເຫດການທີ່ມີຄວາມສຳພັນກັນດ້ວຍເງື່ອນໄຂຕ່າງໆ ເຮັດໃຫ້ຄ່າຄວາມນ່າຈະຜັນປ່ຽນຕາມຄວາມສຳພັນຂອງເຫດການນັ້ນໄດ້ກ່າວຄືຄ່າຄວາມນ່າຈະເປັນໃນແຕ່ລະຕົວປ່ຽນອາດເປັນອິດສະຫຼະຕໍ່ກັນດ້ວຍເງື່ອນໄຂ ຫຼືຄວາມສຳພັນເຫດການໃນຂະນະດຽວກັນອາດມີຕົວປ່ຽນບາງຊຸນິດທີ່ບໍ່ເປັນອິດສະຫຼະຕໍ່ກັນ ແລະ ກັນ ຈຳເປັນຕ້ອງມີການເຊື່ອມຕໍ່ກັບເຫດການອື່ນ ເພື່ອສະແດງເຖິງຄວາມສຳພັນສາມາດອະທິບາຍດ້ວຍການໃຊ້ໂຄງສ້າງຂອງ Bayesian Network ໂດຍມີຄຸນສົມບັດທີ່ສຳຄັນດັ່ງນີ້:

1. ໂນດທັງໝົດໃນ Bayesian Network ແຕ່ລະໂນດຈະແທນດ້ວຍຕົວປ່ຽນຕ່າງໆທີ່ກ່ຽວຂ້ອງກັບເຫດການຫຼືຂໍ້ມູນທີ່ສົນໃຈ
2. ການເຊື່ອມຕໍ່ລະຫວ່າງຄູ່ໂນດດ້ວຍລູກສອນຖ້າລູກສອນອອກຈາກໂນດ X ແລະຊີ້ ຫົວລູກສອນໄປຫາໂນດ Y ຈະເອີ້ນວ່າ “ ໂນດ X ເປັນໂນດພໍ່ແມ່ (Parent) ຂອງໂນດ Y ”
3. ແຕ່ລະໂນດ X ຈະມີເງື່ອນໄຂການກະຈາຍຄວາມນ່າຈະເປັນຄື P(X, I Parents(X,)) ເຊິ່ງຈະສົ່ງຜົນຕໍ່ໂນດພໍ່ແມ່ຂອງແຕ່ລະໂນດ
4. ກຣາບຂອງ Bayesian network ຈະຕ້ອງບໍ່ເຊື່ອມຕໍ່ກັນເປັນວົງຈອນ

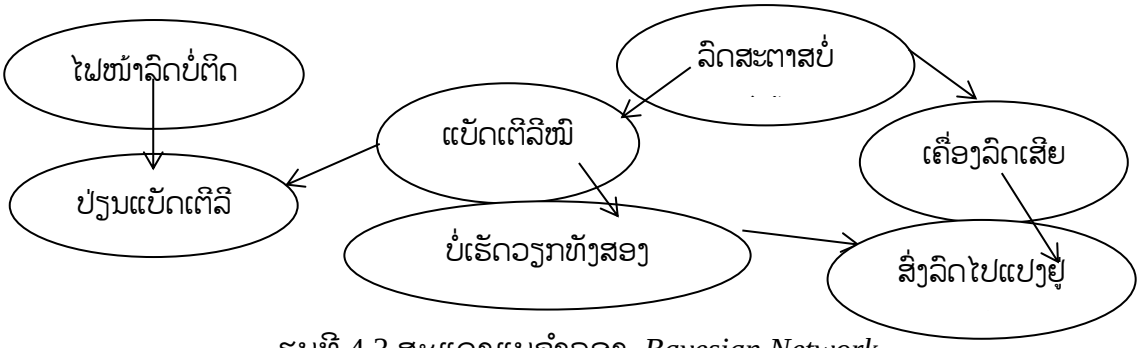
Bayesian Network ຖືກນຳມາໃຊ້ອະທິບາຍຄວາມບໍ່ຂຶ້ນຕໍ່ກັນຢ່າງມີເງື່ອນໄຂ (Condition Independent) ລະຫວ່າງຕົວປ່ຽນ (Variable) ໂດຍໃນໂຄງສ້າງຂອງ Bayesian Network ນີ້ອາດມີໂນດທີ່ເປັນອິດສະຫຼະຈາກໂນດອື່ນລວມຢູ່ນຳກໍ່ໄດ້ພິຈາລະນາຕົວຢ່າງຈາກຮູບທີ 4.2



ຮູບທີ 4.2 ສະແດງໂນດທີ່ເປັນອິດສະຫຼະພາຍໃນ *Bayesian Network*

ຈາກຮູບທີ 4.2 ຈະມີຕົວປ່ຽນທັງໝົດ 4 ຕົວປ່ຽນຄື ສະພາບອາກາດ ລົດສະຕາສບໍ່ໄດ້ ແບບເຕີລີໝົດ ແລະ ເຄື່ອງຈັກລົດເພ່ເຊິ່ງຕົວປ່ຽນສະພາບອາກາດນັ້ນຈະບໍ່ມີຄວາມສຳພັນໃດກັບຕົວປ່ຽນອື່ນເລີຍ ເຮັດໃຫ້ຕົວປ່ຽນສະພາບອາກາດເປັນອິດສະຫຼະບໍ່ຂຶ້ນກັບເງື່ອນໄຂໃດຂອງຕົວປ່ຽນອື່ນພາຍໃນ Bayesian Network ດຽວກັນ ໃນຂະນະທີ່ມີຕົວປ່ຽນສາມຕົວທີ່ສຳພັນກັນນັ່ນຄືຕົວປ່ຽນລົດສະຕາສບໍ່ຕິດເປັນໂນດພໍ່ແມ່ (Parent) ຫຼື ເອີ້ນໄດ້ອີກຢ່າງໜຶ່ງວ່າ “Prior Knowledge” ແລະ ມີຕົວປ່ຽນແບັດເຕີລີກັບເຄື່ອງຈັກລົດເພ່ຢ່າງໃດຢ່າງຫໜຶ່ງ, ຢ່າງໃດກໍ່ຕາມຕົວປ່ຽນທັງສອງບໍ່ຈຳເປັນຕ້ອງເປັນສາເຫດທີ່ເຮັດໃຫ້ເກີດເຫດການລົດສະຕາສບໍ່ຕິດເຫດການດຽວແຕ່ອາດເປັນສາເຫດໃຫ້ເກີດເຫດການອື່ນໄດ້ເຊ່ັນກັນ, ເຊ່ັນວ່າ: ແບັດເຕີລີອາດເປັນສາເຫດທີ່ເຮັດໃຫ້ເກີດໄດ້ຫຼາຍເຫດການ ທັງລົດສະຕາສບໍ່ຕິດແລະໄຟໜ້າລົດບໍ່ຕິດ (ໃນກໍລະນີໄຟໜ້າລົດບໍ່ຕິດຈະສາມາດສະລຸບໄດ້ທັນທີວ່າແບບເຕີລີໝົດ) ເປັນຕົ້ນ.

ການສ້າງແບບຈຳລອງ Bayesian Network ນີ້ຈະໃຊ້ເສ້ັນກຣາບທີ່ມີທິດທາງເພ່ືອເຊ່ືອມລະຫວ່າງໂນດ ໂດຍແຕ່ລະໂນດຈະມີເງ່ືອນໄຈທີ່ສະແດງເຖິງເຫດການຫຼືຄວາມນ່າຈະເປັນຕ່າງໆ ທ່ີອາດເກີດຂ້ຶນກັບຄວາມສຳພັນກັບໂນດເລ່ີມຕ້ົນ ເຊ່ິງເງ່ືອນໄຂທ່ີລະບຸຈະມີຄວາມສຳພັນກັນຕາມເຫດການຕ່າງໆທ່ີສົນໃຈນ້ັນເອງ ເຊ່ັນ ເມ່ືອລົດສະຕາສບໍ່ຕິດ ອາດເກີດຈາກຫຼາຍສາເຫດ ເຄ່ືອງຍົນອາດເກີດຄວາມເສ່ືອຫາຍແບັດເຕີລີໝົດຫຼືອ່ືນໆ ໂດຍແຕ່ລະສາເຫດຈະມີຄວາມຕ່ໍເນ່ືອງກັນ (ດ່ັງຮູບ4.3) ເປັນຕ້ົນ.



ຮູບທີ 4.3 ສະແດງແບຈຳລອງ *Bayesian Network*

ຈາກຮູບທີ 4.3 ຈະເຫັນໄດ້ວ່າເຫດການເລີ່ມຕົ້ນຄືລົດສະຕາສບໍ່ຕິດ, ຈະມີເງື່ອນໄຂຫຼືເຫດການທີ່ກ່ຽວຂ້ອງ 2 ໂນດຄື ເຄື່ອງຈັກລົດເພ່ ແລະ ແບັດເຕີລີໝົດ ແລະ ສະແດງວິທີການແກ້ໄຂບັນຫາຈາກຂໍ້ຜິດພາດດັ່ງກ່າວເຊິ່ງຈະມີທິດທາງແຕກຕ່າງກັນຕາມເຫດການ ຫຼື ເງື່ອນໄຂນັ້ນໂດຍພິຈາລະນາແຕ່ລະສ່ວນວ່າເກີດຈາກເງື່ອນໄຂໃດ, ຫາກແບັດເຕີລີໝົດພຽງຢ່າງດຽວກໍ່ສາມາດແກ້ໄຂໄດ້ໂດຍການປ່ຽນ, ແຕ່ຖ້າເກີດຈາກເຄື່ອງຈັກລົດເພ່ ຫຼື ເປັນທັງສອງຢ່າງຈະຕ້ອງສົ່ງລົດໄປແປງທີ່ອູ່. ນອກຈາກນີ້ເຫດການບາງຢ່າງອາດບົ່ງບອກທີ່ມາຂອງເຫດທີ່ໜ້າສົນໄດ້ເຊັ່ນ: ໄຟໜ້າລົດບໍ່ເຮັດວຽກສະແດງວ່າລະບົບໄຟຟ້າໃນລົດບໍ່ເຮັດວຽກຈຶ່ງສະຫຼຸບໄດ້ທັນທີວ່າແບັດເຕີລີໝົດຢ່າງແນ່ນອນ, ໂດຍບໍ່ຈຳເປັນຕ້ອງວິເຄາະເຫດການອື່ນເຊິ່ງຈະສຳພັນກັບເຫດການທີ່ລົດສະຕາສບໍ່ຕິດເປັນຕົ້ນ.

ຈາກຕົວຢ່າງແບບຈຳລອງນີ້ສົ່ງຜົນໃຫ້ເກີດຂະບວນການຕັດສິນໃຈທີ່ແຕກຕ່າງກັນຂຶ້ນ ເນື່ອງຈາກມີປັດໄຈທີ່ສົ່ງຜົນຕໍ່ຜົນຮັບຫຼາຍກວ່າໜຶ່ງຢ່າງໂດຍສາມາດເລືອກແກ້ໄຂສະຖານະການໄດ້ດັ່ງນີ້:

1. ຫາກລົດສະຕາສບໍ່ໄດ້ ເພາະແບັດເຕີລີໝົດ ເຈົ້າຂອງລົດສາມາດແກ້ໄຂສະຖານະການເບື້ອງຕົ້ນໄດ້ໂດຍການປ່ຽນແບັດເຕີລີ
2. ໃນກໍລະນີໄຟໜ້າລົດບໍ່ຕິດສະແດງວ່າລະບົບໄຟ້ຳໃນລົດບໍ່ເຮັດວຽກຈຶ່ງບໍ່ຕ້ອງວິເຄາະສາເຫດສະພາບແວດລ້ອມອື່ນກ່ອນ ໂດຍສາມາດແກ້ໄຂບັນຫາໄດ້ທັນທີດ້ວຍການປ່ຽນແບັດເຕີລີ
3. ຖ້າເຄື່ອງຈັກມີບັນຫາເຮັດໃຫ້ລົດສະຕາສບໍ່ຕິດຈະຕ້ອງສົ່ງລົດໄປແປງຢູ່ອູ່ເທົ່ານັ້ນ
4. ໃນກໍລະນີທີ່ລົດສະຕາສບໍ່ຕິດເຫດເກີດຈາກເຄື່ອງຈັກເພ່ ແລະ ແບັດເຕີລີໝົດກໍ່ຕ້ອງສົ່ງລົດໄປແປງທີ່ອູ່ເຊັ່ນດຽວກັນ

ຈະເຫັນໄດ້ວ່າການແກ້ໄຂສະຖານະການສາມາດເຮັດໄດ້ຫຼາຍວິທີຂຶ້ນຢູ່ກັບເງື່ອນໄຂຂອງເຫດນັ້ນດ້ວຍວ່າຈະສົ່ງຜົນຕໍ່ການຕັດສິນໃຈແນວໃດ ໂດຍພິຈາລະນາໄດ້ຈາກທິດທາງຂອງລູກສອນທີ່ເຊື່ອມຕໍ່ໄປຫາໂນດຕ່າງໆໃນແບບຈຳລອງໂດຍຫຼັກການຂອງ Bayesian Network ນີ້ຈະສາມາດນຳຮູບແບບຂອງທິດທາງການຕັດສິນໃຈໄປປະຍຸກໃຊ້ກັບປັນຍາປະດິດໄດ້ເປັນຢ່າງດີ. ເນື່ອງຈາກເປັນຮູບແບບການແກ້ໄຂບັນຫາຕາມຫຼັກເຫດຜົນ ແລະ ຄວາມເໝາະສົມຂອງສະຖານະການນັ້ນດ້ວຍ.

4.4.2 ທິດສະດີຂອງ Bayesian Network

ຈາກທີ່ກ່າວມາຈະເຫັນວ່າ Bayesian Network ມີຄຸນສົມບັດຂອງຄວາມສຳພັນໃນແຕ່ລະໂນດທັ້ງ ທີ່ເປັນອິດສະຫຼະຈາກກັນ ແລະ ມີເງື່ອນໄຂຕໍ່ກັນເຮັດໃຫ້ໂນດເຫຼົ່ານີ້ຕ້ອງກ່ຽວຂ້ອງກັບຄວາມສຳພັນທີ່ເອີ້ນ ວ່າ ຄວາມບໍ່ຂຶ້ນຕໍ່ກັນຢ່າງມີເງື່ອນໄຂ (Condition independent) ເຊິ່ງນິຍາຍໄດ້ດັ່ງນີ້

- **ຄວາມບໍ່ຂຶ້ນຕໍ່ກັນຢ່າງມີເງື່ອນໄຂ (Condition Independent)**

X ບໍ່ຂຶ້ນກັບ Y ຢ່າງມີເງື່ອນໄຂກ່າວຄື ຖ້າຄວາມນ່າຈະເປັນຂອງ X ບໍ່ຂຶ້ນກັບຄ່າຂອງ Y ເມື່ອ ຮູ້ຄ່າ Z ແລ້ວຈະຂຽນເປັນສົມຜົນດັ່ງນີ້

$$(\ \forall x_i, y_j, z_k,)\ P(X=x_i \mid Y=y_j, Z=z_k)=P(X=x_i \mid Z=z_k\ )$$

ຫຼື $$P(X \mid Y,Z)=P(X \mid Z)$$

ຈາກສົມຜົນດັ່ງກ່າວໝາຍເຖິງສຳລັບ x_i, y_j, z_k ໃດໆຄວາມນ່າຈະເປັນທີ່ X ຈະມີຄ່າເປັນ x_i ເມື່ອຮູ້ ວ່າ Y ມີຄ່າເປັນ y_j ແລະ Z ມີຄ່າເປັນ z_k ໂດຍມີຄ່າຄວາມນ່າຈະເປັນແມ່ນ X ຈະມີຄ່າເປັນ x_i ເມື່ອຮູ້ ວ່າ Z ມີຄ່າເປັນ z_k ເຊິ່ງຈະອີ້ນວ່າຄ່າຂອງ X ບໍ່ຂຶ້ນກັບຄ່າຂອງ Y ຢ່າງມີເງື່ອນໄຂເມື່ອຮູ້ຄ່າຂອງ Z ດັ່ງນັ້ນຈຶ່ງສາມາດຕັດຕົວປ່ຽນ Y ອອກ ແລະ ຂຽນສົມຜົນໄດ້ເປັນ P (X | Z).

ຄວາມບໍ່ຂຶ້ນຕໍ່ກັນຢ່າງມີເງື່ອນໄຂນີ້ຈະຊ່ວຍໃຫ້ການຫາຄວາມນ່າຈະເປັນຂອງຕົວປ່ຽນທີ່ຕ້ອງການ ງ່າຍຂຶ້ນໂດຍບໍ່ຕ້ອງການສົນໃຈຕົວປ່ຽນອື່ນເຊັ່ນ: ຟ້າຮ້ອງ ຈະບໍ່ຂຶ້ນກັບຝົນຕົກສະເໝີໄປຖ້າຮູ້ວ່າເກີດ ຟ້າແມບເຊິ່ງໝາຍເຖິງຖ້າເກີດເຫດການຟ້າແມບຂຶ້ນສາມາດບອກໄດ້ທັນທີວ່າຈະເກີດສຽງຟ້າຮ້ອງຕາມໃຈ ໂດຍບໍ່ຕ້ອງສົນໃຈວ່າຂະນະນັ້ນຝົນຈະຕົກ ຫຼືບໍ່ ເຮັດໃຫ້ການຫາຄວາມນ່າຈະເປັນງ່າຍຂຶ້ນໂດຍບໍ່ຕ້ອງ ສົນໃຈອີກເຫດການໜຶ່ງຈາກທີ່ກ່າວມາສາມາດຂຽນໄດ້ຄື

$$P(\text{Thunder}/\text{Rain, Lighting})=P(\text{Thunder}/\text{Lighting})$$

ໃນແຕ່ລະໂນດຂອງ Bayesian network ຈຳເປັນຕ້ອງມີຕາຕະລາງຄວາມນ່າຈະເປັນທີ່ເອີ້ນວ່າ ຕາຕະລາງຄວາມນ່າຈະເປັນແບບມີເງື່ອນໄຂ(Conditional Probability Table) ຫຼື ເອີ້ນສັ້ນໆວ່າ ຕາຕະລາງ CPT ເພື່ອໃຊ້ໃນການບົ່ງບອກຄ່າຄວາມນ່າຈະເປັນຂອງໂນດດັ່ງຕົວຢ່າງທີ 4.8

ຕົວຢ່າງທີ 4.8

ສະແດງ Bayesian Network ຂອງຄວາມຄວນທີ່ຈະເປັນສະໜາມຫຍ້າຈະປຽກເນື່ອງຈາກວ່າປັດໄຈ ທາງດ້ານສະພາບອາກາດ ເຊິ່ງມີຕົວປ່ຽນທັງໝົດ 4 ຕົວປ່ຽນ ໂດຍກຳນົດໄວ້ດັ່ງນີ້:

ມີເມກຫຼາຍ = C (Cloudy) ເປີດ Sprinkler = S (Sprinkler) ຝົນຕົກ = R (Rain) ສະໜາມ ຫຍ້າປຽກ = W (Wet Grass)

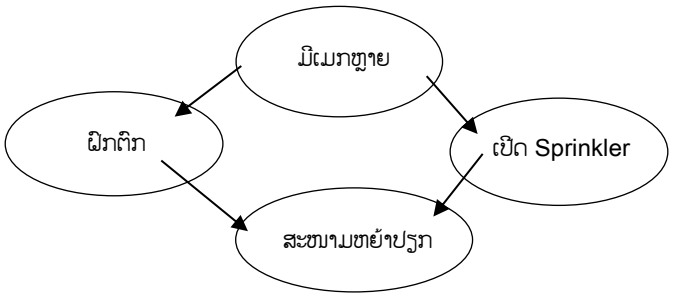
ໂດຍຈະສະແດງຄວາມສຳພັນລະຫວ່າງເຫດການທີ່ເຮັດໃຫ້ສະໜາມຫຍ້າປຽກ ເຊິ່ງພິຈາລະນາ ຈາກຄວາມທີ່ອາດຈະເປັນສາເຫດທີ່ເຮັດໃຫ້ສະໜາມຫຍ້າປຽກ ມີເຫດການທີ່ກ່ຽວຂ້ອງຂອງ 2 ເຫດການ ຄື ຝົນຕົກ ແລະ ມີການເປີດ Sprinkler ຈາກທັງສອງເຫດການໂອກາດທີ່ຈະເກີດຂຶ້ນໃນເວລາປົກກະຕິຈະ ແຕກຕ່າງກັນໂດຍສິ້ນເຊີງ ເນື່ອງຈາກຝົນຕົກໃນສະພາບອາກາດປົກກະຕິເປັນໄປໄດ້ຍ່ອນຂ້າງຍາກ ແລະ

ການເປີດ Sprinkler ກໍສາມາດເປີດໄດ້ຕະຫຼອດເວລາ ເຮັດໃຫ້ຕ້ອງພິຈາລະນາເຖິງປັດໄຈຕົ້ນເຫດທີ່ແທ້ຈິງ ໃນທີ່ນີ້ຄື ສະພາບອາກາດ.

ເນື່ອງຈາກສະພາບອາກາດມີເມກຫຼາຍມີຄວາມເປັນໄປໄດ້ວ່າທີ່ຈະມີຝົນຕົກໄດ້ຄ່ອນຂ້າງຫຼາຍ ເຊິ່ງອາດຈະເປັນສາເຫດຫຼັກທີ່ເຮັດໃຫ້ສະໜາມຫຍ້າປຽກ ແລະ ໃນບາງໂອກາດອາດຈະເກີດການເປີດ Sprinkler ກໍໄດ້ ແຕ່ການຈະເປີດ Sprinkler ເພື່ອຫົດນ້ຳສະໜາມຫຍ້າໃນສະພາບອາກາດຟ້າມືດນັ້ນເປັນໄປໄດ້ຍາກ ຈຶ່ງມີຄ່າຄວາມນ່າຈະເປັນຄ່ອນຂ້າງນ້ອຍ.

❖ ສະແດງຄວາມສຳພັນຂອງຕົວປ່ຽນ ແລະ ຕາຕາລສງ CPT ດັ່ງຮູບທີ 4.4

P(C) = 0.5



C	P(R)
T	0.8
F	0.2

C	P(S)
T	0.1
F	0.5

R	S	P(W)
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.00

ຮູບທີ 4.4 ສະແດງຕົວຢ່າງ Bayesian Network ແລະ ຕາຕາຕະລາງ CPT[Russell and Nerving, 2003]

ຮູບທີ 4.4 ສະແດງຄວາມນ່າຈະເປັນຂອງແຕ່ລະເຫດການ ໄວ້ໃນຕາຕາຕະລາງ CPT ເຊິ່ງອະທິບາຍໄດ້ວ່າ ໃນສະພາວະປົກກະຕິໂອກາດທີ່ສະພາບອາກາດຈະມີເມກຫຼາຍນັ້ນມີຄວາມນ່າຈະເປັນ ຄື P(C) = 0.5 ແລະເມື່ອສະພາບອາດກາດເປັນລັກສະນະດັ່ງກ່າວ ຈະມີເຫດການທີ່ສາມາດເກີດຂຶ້ນໄດ້ 2 ຢ່າງ ຄື ເກີດຝົນຕົກ ແລະ ເປີດ Sprinkler ໂດຍທັງສອງຢ່າງຈະມີຄວາມນ່າຈະເປັນການເປີດ Sprinkler ຈະມີຄ່ອນຂ້າງຕ່ຳຄື P(S) = 0.1 ເນື່ອງຈາກມີໂອກາດສູງທີ່ຝົນຈະຕົກເຮັດໃຫ້ມີຄວາມຈຳເປັນທີ່ຈະເປີດ Sprinkler ລົດລົງນັ້ນເອງ.

ໃນຂະນະທີ່ສະໜາມຫຍ້າປຽກຈະເກີດຈາກ 2 ເຫດການ ດັ່ງນັ້ນຈຶ່ງຕ້ອງພິຈາລະນາຄວາມອາດຈະເປັນຈາກສອງເຫດການດັ່ງກ່າວເຊິ່ງຜົນທີ່ໄດ້ເປັນດັ່ງຕາຕາຕະລາງຄ່າຄວາມອາດຈະເປັນທີ່ສະແດງໄວ້ໃນຕົວຢ່າງ ເຮັດໃຫ້ຮູ້ໄດ້ວ່າໂອກາດທີ່ສະໜາມຫຍ້າຈະປຽກນັ້ນມີຫຼາຍເຖິງ 0.9 ໃນທາງກັບກັນໂອກາດທີ່ທັງສອງເຫດການຈະບໍ່ເກີດຂຶ້ນ ແລະ ເຮັດໃຫ້ສະໜາມຫຍ້າບໍ່ປຽກກໍມີເຊັ່ນດຽວກັນ.

- ຄວາມອາດຈະເປັນຮ່ວມ (Joint Portability)

ໃນ Bayesian Network ເປັນຕົວປ່ຽນແຕ່ລະໂຕຈະມີຄ່າຄວາມອາດຈະເປັນສະເພາະທີ່ອາດເປັນຄວາມອາດຈະເປັນຂອງໂນດໂຕເລີ່ມຕົ້ນ ຫຼື ຄວາມອາດຈະເປັນທີ່ໄດ້ຈາກຄວາມສຳພັນຫຼາຍກ່ວາໜຶ່ງໂນດ ໂດຍຄວາມນອາດຈະເປັນທີ່ມາຈາກຕົວປ່ຽນຫຼາຍກ່ວາໜຶ່ງໂຕເອີ້ນວ່າ “ຄວາມອາດຈະເປັນຮ່ວມ (Joint Portability)”ດັ່ງນີ້:

$$P(x_1, x_2, \dots, x_n) = \prod_{i=1}^{n} P(X_i | Parents(X_i))$$

Parents (x_i) ໝາຍເຖິງ ໂນດພໍ່ແມ່ໂດຍກົງຂອງ X_i ຈາກສົມຜົນເບື້ອງຕົ້ນສາມາດຫາຄ່າຄວາມອາດຈະເປັນທີ່ $X_1, X_2, \dots, X_n$ ເກີດຂຶ້ນພ້ອມກັນ ໂດຍນຳຄ່າຄວາມອາດຈະເປັນຂອງແຕ່ລະໂນດຄູ່ກັນ ເຊິ່ງຕ້ອງພິຈາລະນາວ່າແຕ່ລະໂນດຂຶ້ນກົງກັບໂນດພໍ່ແມ່ໂນດໃດດ້ວຍ ດັ່ງຕົວຢ່າງທີ 4.9

ຕົວຢ່າງທີ 4.9

ຈາກຕົວຢ່າງທີ 4.8 ຈະສະແດງລາຍລະອຽດຂອງຄວາມອາດຈະເປັນໃນແຕ່ລະຕົວປ່ຽນ ຫາກຕ້ອງການທີ່ຈະຮູ້ວ່າຄວາມອາດຈະເປັນທີ່ສະໜາມຫຍ້າຈະປຽກເພາະຝົນຕົກໜຽງຢ່າງດຽວ ໂດຍສະພາບອາດກາດໃນຕອນນັ້ນບໍ່ມີເມກຫຼາຍທີ່ຈະສົ່ງຜົນເຮັດໃຫ້ຝົນຕົກໄດ້.

ໂດຍພິຈາລະນະຈາກຄ່າຄວາມອາດຈະເປັນໃນຕາຕາຕະລາງ CPT ຂອງແຕ່ລະຕົວປ່ຽນ ຕາມຄວາມຕ້ອງການທີ່ກຳນົດຂຶ້ນ

ຄື ສະໜາມຫຍ້າໂດຍສະພາບອາກາດບໍ່ມີເມກດຳຫຼາຍ

ແທນຄ່າລົງໃນສົມຜົນຄວາມອາດຈະເປັນຮ່ວມ ໄດ້ດັ່ງນີ້:

$P(W \wedge S \wedge R \wedge \neg C) = P(W|S \wedge R)P(S|\neg C)P(R|\neg C)P(\neg C)$

$= 0.99 \times 0.5 \times 0.2 \times (1-0.5)$

$= 0.99 \times 0.5 \times 0.2 \times 0.5$

$= 0.0495$

ດັ່ງນັ້ນ ຈຶ່ງໄດ້ຄ່າຄວາມອາດຈະເປັນຮ່ວມຂອງສະໜາມຫຍ້າປຽກໂດຍອາກາດບໍ່ມີເມກຫຼາຍ ຄື 0.0495

ຈາກຕົວຢ່າງທີ 4.9 ເມື່ອແທນຄ່າແລ້ວຈະໄດ້ຜົນລັດດັ່ງທີ່ກ່າວມາຂ້າງຕົ້ນ ເຊິ່ງຈະເຫັນໄດ້ວ່າຕົວປ່ຽນບາງໂຕອາດບໍ່ຂຶ້ນກັບໂຕໃດເລີຍ ໃນຂະນະທີ່ບາງໂຕຂຶ້ນກົງກັບຕົວປ່ຽນອື່ນຫຼາຍກ່ວາຕົວປ່ຽນໃດໜຶ່ງ ຫາກພິຈາລະນະຕາມສົມຜົນຫາຄວາມອາດຈະເປັນຮ່ວມແລ້ວ, ຈຳເປັນຕ້ອງມີການຫຼຸດຮູບແບບ ໃຫ້ສອດຄ່ອງກັບຄວາມສຳພັນຂອງແຕ່ລະໂນດນຳ. ໂນດໃດທີ່ບໍ່ຂຶ້ນຕໍ່ກັນກໍ່ບໍ່ຈຳເປັນຕ້ອງຄົງຮູບແບບໄວ້ ເພື່ອສະແດງໃຫ້ເຫັນຄວາມສຳພັນຂອງແຕ່ລະໂນດໄດ້ຢ່າງຈະແຈ້ງ. ຈາກທີ່ກ່າວວ່າການຫາຄ່າຄວາມອາດຈະເປັນຮ່ວມຈະຕ້ອງຮູ້ຄ່າຂອງຄວາມອາດຈະເປັນຂອງທຸກໂນດ ແຕ່ໃນກໍລະນີທີ່ຮູ້ຄ່າຄວາມອາດຈະເປັນພຽງບາງໂນດຈຳເປັນຕ້ອງໃຊ້ເຕັກນິກອື່ນເຂົ້າມາຊ່ວຍນັ້ນແມ່ນ ເຕັກນິກການອະນຸມານທີ່ໃຊ້ສຳລັບ Bayesian Network ເຊິ່ງຈະກ່າວລາຍລະອຽດໃນຫົວຂໍ້ຕໍ່ໄປ.

4.4.3 ການອະນຸມານຂອງ Bayesian Network

ການອານຸມານຂອງ Bayesian Network ຈະໃຊ້ໃນການຫາຄ່າຄວາມອາດຈະເປັນຂອງຕົວປ່ຽນຫຼືໂນດທີ່ເຮົາສົນໃຈ ເພື່ອໃຊ້ໃນການສະໜັບສະໜູນວິທີ່ອື່ນໆ ຫຼືຊ່ວຍໃຫ້ສາມາດຄຳນວນຫາຜົນຮັບໄດ້ຢ່າງສົມບູນ ເນື່ອງຈາກວ່າໃນການຄຳນວນຈຳເປັນຕ້ອງຮູ້ຄ່າຄວາມອາດຈະເປັນຂອງທຸກໂນດ ເຕັກນິກການອານຸມານຂອງ Bayesian Network ມີດັ່ງນີ້

1. **ການອານຸມານຈາກເຫດ (Casual Reasoning)**

ເປັນການອານຸມານດ້ວຍເຫດຜົນເພື່ອໃຊ້ຫາຄ່າຄວາມອາດຈະເປັນຜົນທີ່ເກີດຂຶ້ນ ເຊິ່ງເຮົາຈະຮັບຊາບຄວາມອາດຈະເປັນຂອງຕົວປ່ຽນທີ່ເປັນເຫດ ແລະ ຕ້ອງການຮັບຊາບຄ່າຄວາມອາດຈະເປັນຂອງຕົວປ່ຽນທີ່ເປັນຜົນ ໃນບາງກໍລະນີອາດບໍ່ສາມາດຫາຄ່າຄວາມອາດຈະເປັນຂອງຜົນໄດ້ໂດຍກົງຈຶ່ງຕ້ອງດຳເນີນການພິຈາລະນາຕັ້ງແຕ່ໂນດພໍ່ແມ່ ທີ່ເປັນເຫດທີ່ເກີດຜົນຕາມທີ່ເຮົາສົນໃຈ.

2. **ການອານຸມານຈາກຜົນ (Diagnosis Reasoning)**

ເປັນການອານຸມານທີ່ນຳຄ່າຄວາມອາດຈະເປັນຂອງຜົນມາເພື່ອຫາຄ່າຄວາມອາດຈະເປັນຂອງເຫດຕາມທີ່ເຮົາສົນໃຈ ເຊິ່ງກົງກັນຂ້າມກັບວິທີທຳອິດ ໃນບາງກໍລະນີການຫາຄ່າຕົວປ່ຽນທີ່ເປັນເຫດ ຫຼື ໂນດພໍ່ແມ່ອາດຈະເຮັດຍາກ ຈຶ່ງຈຳເປັນຕ້ອງອາໄສ ໂນດລູກທີ່ເປັນຜົນເພື່ອສະໜັບສະໜູນໃຫ້ການຄຳນວນງ່າຍຂຶ້ນ.

3. **ການອະທິບາຍເພື່ອລົດຄວາມເປັນໄປໄດ້ (Explaining Away)**

ເປັນການອານຸມານທີ່ປະສົມລະຫວ່າງສອງວິທີ່ທຳອິດຄື ການອານຸມານຈາກເຫດ ແລະ ອະນຸມານຈາກຜົນ ໂດຍຈະໃຊ້ວິທີການນີ້ເພື່ອຄົ້ນຫາຄ່າຄວາມອາດຈະເປັນຂອງຕົວປ່ຽນທີ່ນ່າສົນໃຈ ເຊິ່ງພິຈາລະນາຫາຄ່າຄວາມອາດຈະເປັນຂອງຕົວປ່ຽນຫຼືໂນດທີ່ມີຄວາມສຳພັນ ແລະ ກ່ຽວຂ້ອງກັນ ເພື່ອພິຈາລະນາລົດຄ່າຄວາມອາດຈະເປັນຕົວປ່ຽນດັ່ງກ່າວ ວ່າມີຄວາມເປັນໄປໄດ້ທີ່ຈະເກີດຂຶ້ນນ້ອຍລົງຫຼືບໍ່?

ການອານຸມານແບບຕ່າງໆ ຈະຊ່ວຍໃຫ້ການຫາຄ່າຄວາມອາດຈະເປັນເຮັດໄດ້ງ່າຍຂຶ້ນ ເນື່ອງຈາກສະຖານະການທີ່ແຕກຕ່າງກັນອາດເຮັດໃຫ້ບໍ່ຮັບຊາບຄ່າຂອງຄວາມອາດຈະເປັນຂອງເຫດການທັງໝົດ ດ້ວຍເຫດນີ້ຈຶ່ງພິຈາລະນາເຕັກນິກຂອງການອານຸມານທີ່ເໝາະສົມກັບສິ່ງທີ່ສົນໃຈ ເພື່ອໃຫ້ຄ່າຄວາມອາດຈະເປັນທີ່ຕ້ອງການຢ່າງແທ້ຈິງ.

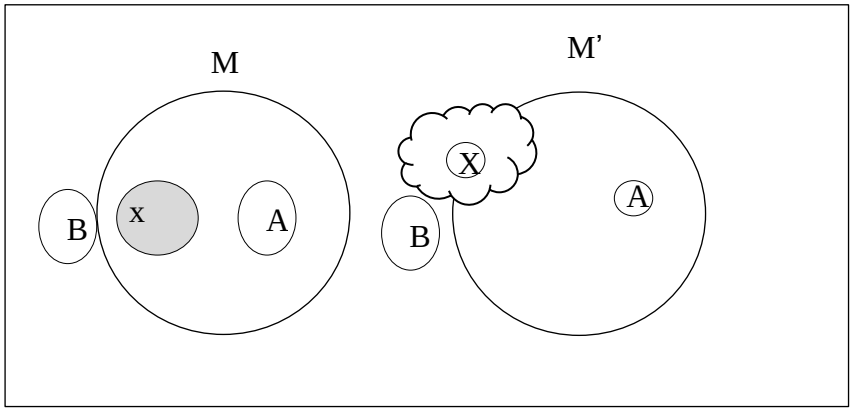
4.5 Fuzzy Logic

Fuzzy Logic ເປັນວິທີທາງຄະນິດສາດ, ເຊິ່ງໃຊ້ເຊື່ອມຕໍ່ກັບຂະບວນການຄຳນວນຂອງລະບົບທີ່ມີການຄຳນວນແຕກຕ່າງກັນ ໂດຍລະບົບໜຶ່ງອາດຄຳນວນແຕກຕ່າງກັນ ໂດຍລະບົບໜຶ່ງອາດຄຳນວນຕາມຫຼັກການທາງຄະນິດສາດ, ສ່ວນອີກລະບົບໜຶ່ງອາດຄຳນວນດ້ວຍຫຼັກການທີ່ບໍ່ມີກົດເກນທີ່ຈະແຈ້ງທຽບຄືກັບການເຮັດວຽກລະຫວ່າງສະໝອງກັບເຄື່ອງຄອມພິວເຕີ, ເຊິ່ງມີຄວາມແຕກຕ່າງກັນໃນລະບົບການຄິດ ແລະ ເຮັດວຽກ, ການຄິດຂອງລະບົບຄອມພິວເຕີຈະມີຂັ້ນຕອນ ແລະ ວິທີການທີ່ສາມາດເຂົ້າໃຈໄດ້ຢ່າງຈະແຈ້ງ, ໃນຂະນະທີ່ສະໝອງມີລັກສະນະການເຮັດວຽກຄ່ອນຂ້າງຊັບຊ້ອນ, ມີປັດໃຈ ແລະ ຕົວປ່ຽນຈຳນວນຫຼາຍ ແລະ ມີຄວາມບໍ່ແນ່ນອນສູງກ່ອນ. ດ້ວຍເຫດນີ້ທິດສະດີຂອງ Fuzzy Logic ຈຶ່ງນຳໄປໃຊ້ກັບລະບົບທີ່ມີຄວາມແປປວນໃນຂະບວນການດຳເນີນງານຄ່ອນຂ້າງສູງ ຫຼື ລະບົບທີ່ໄດ້ຜົນຮັບບໍ່ແນ່ນອນ.

ເນື່ອງຈາກ Fuzzy Logic ເປັນຫຼັກການທາງຄະນິດສາດທີ່ກ່ຽວຂ້ອງກັບຄວາມບໍ່ແນ່ນອນ ໂດຍທິດສະດີທາງຄະນິດສາດສ່ວນຫຼາຍມີຈະນຳມາໃຊ້ກ່ຽວກັບຄວາມອາດຈະເປັນ ແລະ ເຊັດ (Set Theory) ດ້ວຍເຫດນີ້ Fuzzy Logic ໄດ້ຖືກນຳປະຍຸກໃຊ້ກັບລະບົບປັນຍາປະດິດຕ່າງໆ ເຊິ່ງເນັ້ນໃນການຮຽນອອກແບບລະບົບຄວາມຄິດຂອງສະໝອງມະນຸດ Fuzzy Logic ໄດ້ຖືກພັດທະນາຂຶ້ນ ແລະ ເປັນທີ່ຮູ້ຈັກຄັ້ງທຳອິດເມື່ອປີ ຄ.ສ 1964-1965 ໂດຍ Lotfi Zedeh ຖືກນຳມາໃຊ້ເພື່ອແກ້ໄຂບັນຫາຂະບວນການຄວາມຄິດຂອງມະນຸດທີ່ຄວບຄຸມລະບົບ ໃຫ້ມີຫຼັກການ ແລະ ມາດຕະການທີ່ຈະແຈ້ງເພື່ອນຳໄປປະຍຸກໃຊ້ຕໍ່ [Kerkira, 2005].

ແນວຄິດຂອງ Fuzzy Logic ຄື ກຳນົດຄ່າຄວາມອາດເປັນລົງໃນຕົວປ່ຽນ ເພື່ອໃຫ້ຕົວປ່ຽນສາມາດສະແດງຄວາມໝາຍອອກມາໄດ້ຢ່າງຈະແຈ້ງ, ເຊັ່ນ: ຜ້າປຸກ ເປັນຄຳທີ່ລະບຸຄວາມຊື້ນຂອງຄ່າທີ່ປຸກນ້ຳ ແຕ່ກໍບໍ່ສາມາດລະບຸໄດ້ວ່າຜ້າຜືນນີ້ປຸກນ້ຳໜ້ອຍ, ຫຼາຍເທົ່າໃດ ດັ່ງນັ້ນ, ຈຶ່ງມີການລະບຸເພື່ອເພີ່ມນ້ຳໜັກກັບຄຳວ່າ “ປຸກ” ໃນທາງກັບກັນກໍຕ້ອງລະບຸຄ່າຂອງຜ້າທີ່ເກືອບແຫ້ງຫຼືປຸກພຽງເລັກນ້ອຍດ້ວຍ ເພື່ອສະແດງໃຫ້ເຫັນຄວາມແຕກຕ່າງຢ່າງຈະແຈ້ງ ການໃຊ້ຄຳໃນພາສາບາງຄັ້ງບໍ່ສາມາດທີ່ຈະລະບຸຄວາມແນ່ນອນຂອງສິ່ງທີ່ຕ້ອງການສື່ອອກມາໄດ້ຢ່າງຈະແຈ້ງ ດັ່ງນັ້ນ, ຈຶ່ງຈຳເປັນຕ້ອງນຳ Fuzzy Logic ເຂົ້າມາມີສ່ວນຮ່ວມ ຕົວຢ່າງຄຳທີ່ໃຊ້ໃນການລະບຸນ້ຳໜັກ ເຊັ່ນ: ໃກ້, ໄກ ແລະ ນ້ອຍເປັນຕົ້ນ.

ແນວຄິດຂອງ Fuzzy Logic ສະແດງດັ່ງຮູບ 4.5



Membership Value of A = 1

Membership Value of B = 0

Membership Value of X = 1

Membership Value of A = 1

Membership Value of B = 0

Membership Value of X: 0<X<1

ຮູບທີ 4.5 ສະແດງແນວຄິດຂອງ Fuzzy Set ແລະ Membership Value [Akerkar, 2005]

ຈາກຮູບທີ 4.5 ເປັນການນິຍາມການກຳນົດຄ່າຂອງ Membership Value ໃຫ້ກັບຂໍ້ມູນທີ່ສົນໃຈ ເພື່ອໃຫ້ຂໍ້ມູນດັ່ງກ່າວມີຄ່າທີ່ໃກ້ຄຽງກັບຄວາມຈິງຫຼາຍທີ່ສຸດ ໂດຍຈະສະແດງໃຫ້ເຫັນຄ່າຂອງ X ຈະບໍ່ມີພຽງຄ່າດຽວ ການລະບຸຊ່ວງໃຫ້ກັບຄ່າຂອງ X ຈະຊ່ວຍໃຫ້ການກຳນົດຄ່າດີຂຶ້ນ.

ຕົວຢ່າງການໃຫ້ຄ່ານ້ຳໜັກໃຫ້ສິ່ງທີ່ສົນໃຈ ເພື່ອລະບຸຄວາມຈະແຈ້ງຂອງເຫດການ ສະແດງຕາ ຕາຕະລາງທີ 4.5

ຄຳອະທິບາຍ	Membership Value
ແຫ້ງຫຼາຍ(Very Dry)	0.1
ແຫ້ງ(Dry)	0.3
ປານກາງ(Average)	0.5
ປຽກ(Wet)	0.7
ປຽກຫຼາຍ(Very Wet)	0.9

ຕາຕາຕະລາງທີ 4.5 ສະແດງຕົວຢ່າງການກຳນົດຄ່າ Membership Value ຕາມລຳດັບຂໍ້ມູນທີ່ສົນໃຈ

ຈາກຕາຕາຕະລາງ 4.5 ຈະເຫັນໄດ້ຄ່າຄວາມຊື້ນ ແລະ ຄວາມແຫ້ງແຕ່ລະປະເພດຈະຖືກກຳນົດ ຄ່າ Membership Value ເພື່ອໃຫ້ເຄື່ອງຄອມພິວເຕີສາມາດລະບຸສະຖານະຂອງເຫດການທີ່ຕ້ອງປະເຊີນ ໄດ້ຢ່າງຈະແຈ້ງຂຶ້ນໄດ້ຈາກການຄຸມເຄືອທີ່ຕ້ອງປະເຊີນ ຫຼື ຕ້ອງເຈີກັບບັນຫາທີ່ບໍ່ສາມາດແກ້ໄຂໄດ້ໂດຍສົມ ເຫດ ຫຼື ຫຼັກການທີ່ມີກົດຈະແຈ້ງຕາຍຕົວ ຫຼັກການ Fuzzy Logic ຈະຊ່ວຍໃຫ້ຂໍ້ມູນມີການຢືດຢຸ່ນຫຼາຍຂຶ້ນ ແລະ ຊ່ວຍໃຫ້ຕັດສິນໃຈໄດ້ໄວຂຶ້ນ.

ຫຼັກການຂອງ Fuzzy Logic ມັກຈະກ່ຽວຂ້ອງ ແລະ ເປັນຫຼັກການສຳຄັນທີ່ໃຊ້ກັບຂໍ້ມູນທີ່ມີຄ່າບໍ່ ແນ່ນອນ ທີ່ສົ່ງຜົນໃຫ້ບໍ່ສາມາດລະບຸຄ່າຈະແຈ້ງໄດ້ ພາສາທຳມະຊາດ (Natural Language) ເປັນອີກໜຶ່ງ ປະເພດທີ່ມີຄຸນສົມບັດຄ່າຂອງຄວາມບໍ່ແນ່ນອນ ສະແດງໃຫ້ເຫັນໄດ້ຢ່າງຈະແຈ້ງ ເນື່ອງຈາກຄຳໃນແຕ່ລະ ພາສາບໍ່ສາມາດລະບຸຄ່າ ຫຼື ຄວາມໝາຍທີ່ບົ່ງບອກປະລິມານທີ່ແນ່ນອນໄດ້ ດັ່ງຕົວຢ່າງທີ່ໄດ້ກ່າວມາຂ້າງ ເທິງ ນອກຈາກທີ່ໄດ້ກ່າວມາໃນຕາຕາຕະລາງທີ 4.5 ຍັງມີຄຳອື່ນໆເຊັ່ນ: ລະດັບຄວາມສູງ,ລະດັບຄວາມ ຮ້ອນ,ຄວາມໜາ,ຄວາມໜັກ ແລະ ທຳມະຊາດ ເຊິ່ງຄຳເຫຼົ່ານີ້ມັກພົບຈາກປະໂຫຍກຫຼືຄຳເວົ້າໃນທຳມະ ຊາດ. ຕົວຢ່າງຂອງພາສາທຳມະຊາດສະແດງດັ່ງນີ້:

“ສຸພົນເປັນຄົນສູງ” ເປັນປະໂຫຍກທີ່ບໍ່ສາມາດລະບຸເຖິງລະດັບຄວາມສູງທີ່ຈະແຈ້ງໄດ້

“ມື້ນີ້ອາດຮ້ອນ” ເປັນປະໂຫຍກທີ່ບໍ່ສາມາດລະບຸໄດ້ວ່າອາກາດຮ້ອນລະດັບໃດ

“ຫ້ອງປົດສອບເສັງວິຊານີ້ຍາກ” ເປັນປະໂຫຍກທີ່ສະແດງເຖິງຄວາມຍາກຂອງຫ້ອງປົດສອບເສັງ ແຕ່ ບໍ່ຮູ້ວ່າມັນຫຼາຍເທົ່າໃດ

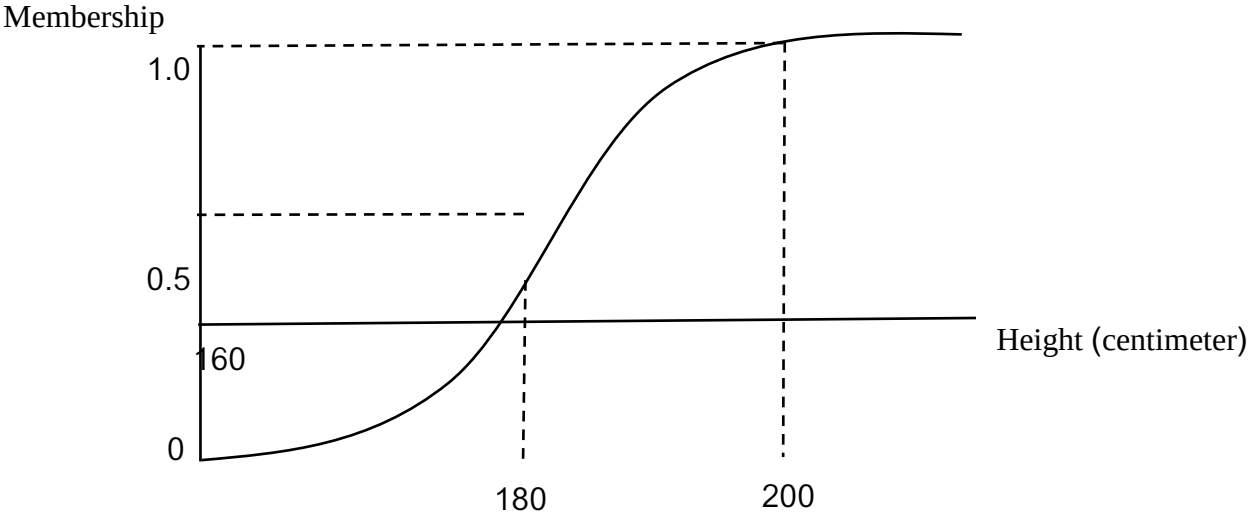
ຈາກປະໂຫຍກທີ່ກ່າວມາຂ້າງເທິງນີ້ ນອກຈາກບໍ່ສາມາດລະບຸຄ່າແນ່ນອນຂອງຄຳຄຸນນາມໄດ້ແລ້ວ ຍັງບໍ່ຮູ້ແດ່ວ່າປະໂຫຍກທີ່ກ່າວມານີ້ເປັນຈິງ ຫຼື ເຊື່ອໄດ້ຫຼາຍ,ນ້ອຍເທົ່າໃດ ເນື່ອງຈາກບໍ່ມາມີມາດຕະການ ຫຼື ການກຳນົດລະດັບຂອງຄຳຄຸນນາມເຫຼົ່ານັ້ນໄດ້ຢ່າງຈະແຈ້ງ ບາງກຸ່ມຄົນອາດເຫັນດ້ວຍ ຫຼື ຄິດວ່າເປັນ ຈິງ, ແຕ່ໃນບາງກຸ່ມຄົນອາດບໍ່ເຫັນດ້ວຍ ທັ້ງນີ້ເພາະທັດສະນະຄະຕິ ແລະ ການກຳນົດມາດຕະຖານທີ່

ແຕກຕ່າງກັນ ດັ່ງນັ້ນການກຳນົດຄ່າລະດັບ ຫຼື Membership Value ທີ່ກ່າວໄວ້ໃນຂ້າງເທິງຈຶ່ງມີຄວາມສຳຄັນຫຼາຍ ເພື່ອໃຫ້ຄຳຄຸນນາມເຫຼົ່ານີ້ສະແດງຄວາມໝາຍໄດ້ຖືກຕ້ອງໄດ້ ໂດຍການໃຊ້ຄຳສະແດງເຖິງນ້ຳໜາຂອງສິ່ງທີ່ສົນໃຈເຊັ່ນ: ຫຼາຍ,ປານກາງ ແລະ ນ້ອຍ ເປັນຕົ້ນ. ເຊິ່ງຄຳເຫຼົ່ານີ້ຈະຊ່ວຍຂະຫຍາຍຄວາມໝາຍທີ່ແນ່ນອນຂອງສິ່ງທີ່ສົນໃຈໄດ້ຫຼາຍຂຶ້ນເຊັ່ນ: ສຸພົນເປັນຄົນສູງຫຼາຍ ຫຼື ຫົວບົດສອບເສັງວິຊາມີ້ຍາກທີ່ສຸດ ເປັນຕົ້ນ.ຈາກທີ່ກ່າວມາຈະຊ່ວຍໃຫ້ເຂົ້າໃຈເຖິງນ້ຳໜັກ ແລະ ລະດັບຂອງຄຳຄຸນນາມນີ້ຫຼາຍຂຶ້ນ ໃນ Fuzzy Logic ຈຶ່ງມີການກຳນົດຄ່ານ້ຳໜັກ ເພື່ອໃຊ້ແກ້ບັນຫາຄວາມບໍ່ແນ່ນອນທີ່ເກີດຂຶ້ນໃນທຳມະຊາດດັ່ງຕົວຢ່າງຕໍ່ໄປນີ້:

ຕົວຢ່າງທີ 4.10

"ສຸພົນເປັນຄົນສູງ"

ໃນຄວາມເປັນຈິງສຸພົນສູງ 180 ຊັງຕີແມດ ຈາກປະໂຫຍກດັ່ງກ່າວສະແດງໃຫ້ເຫັນວ່າສຸພົນເປັນຄົນສູງ ແຕ່ບໍ່ສາມາດລະບຸໄດ້ວ່າສູງໃນລະດັບໃດ,ເຊິ່ງປະໂຫຍກດັ່ງກ່າວອາດຈະເກີດຈາກມຸມມອງຂອງບຸກຄົນທົ່ວໄປ ຫາກມີການກຳນົດຄ່ານ້ຳໜັກ ຫຼື Membership Function ໃຫ້ ໂດຍປຽບທຽບກັບຄ່ານ້ຳໜັກຂອງກຸ່ມຄົນທົ່ວໄປ ຈະໄດ້ດັ່ງຮູບທີ 4.6



ຮູບທີ 4.6 ກຣາຟຄວາມສຳພັນຂອງ Membership Function ກັບລະດັບຄວາມສູງ

ຈາກຮູບທີ 4.6 ຈະສະແດງໃຫ້ເຫັນຄ່າ Membership Function ຂອງກຣາຟຄວາມສູງສະເລ່ຍຂອງບຸກຄົນທົ່ວໄປຢູ່ໃນຊ່ວງ 160-200 ຊ.ມ ເຊິ່ງຄ່ານ້ຳໜັກຈະເພີ່ມຂຶ້ນຕາມລະດັບຄວາມສູງ ຫາກບຸກຄົນໃດຫາກມີຄວາມສູງນ້ອຍກ່ວາ 160 ຊ.ມ ຈະຖືວ່າເປັນຄົນບໍ່ສູງ ໃນຂະນະດຽວກັນກັບບຸກຄົນໃດທີ່ມີຄວາມສູງເກີນ 200 ຊ.ມ ຈະຖືກວ່າເປັນຄົນສູງທັງໝົດ ສຸພົນສູງ 180 ຊ.ມ ຈຶ່ງມີຄ່າ Membership Function ເປັນ 0.5 ຖ້າມີຄວາມສູງ 200 ຊ.ມ ຈະມີຄ່ານ້ຳໜັກເປັນ 1.0 ຈາກທີ່ກ່າວມາ ຈຶ່ງເຮັດໃຫ້ສາມາດລະບຸຄ່າຂອງຄວາມບໍ່ແນ່ນອນໃນພາສາທຳມະຊາດໃນລະດັບໜຶ່ງ, ເຊິ່ງຊ່ວຍໃຫ້ສາມາດຈຳແນກຄວາມສຳຄັນຂອງຂໍ້ມູນທີ່ສົນໃຈໄດ້ຢ່າງຈະແຈ້ງຫຼາຍຂຶ້ນ. ໃນນີ້ຂຶ້ນຢູ່ກັບຄວາມເຫັນຂອງບຸກຄົນທົ່ວໄປວ່າຄ່າຄວາມສູງລະດັບໃດທີ່ເອີ້ນວ່າ "ເປັນຄົນສູງ" ໃນທາງກົງກັນຂ້າມຖ້າຫາກວ່າເປັນການສຳຫຼວດຈາກກຸ່ມ

ນັກກິລາບານບ່ວງທີ່ມີຄວາມສູງສະເລ່ຍຄ່ອນຂ້າງຫຼາຍ, ການກຳນົດຊ່ວງລະດັບຄວາມສູງຈະແຕກຕ່າງຈາກເກົ່າຢ່າງຈະແຈ້ງ, ໂດຍກຸ່ມນັກກິລາບານບ່ວງອາດກຳນົດຊ່ວງເປັນ 170-210 ຊ.ມ, ເຊິ່ງເຮັດໃຫ້ຄ່ານ້ຳໜັກຂອງຄວາມສູງຂອງສຸພົນຢູ່ໃນລະດັບທີ່ແຕກຕ່າງຈາກເກົ່າຢ່າງແນ່ນອນ ດັ່ງນັ້ນຈຶ່ງຕອ້ງຄຳນຶງເຖິງການກຳນົດຄ່ານ້ຳໜັກດ້ວຍ ເພື່ອຄວາມສອດຄ່ອງ ກັບຂໍ້ມູນທີ່ເຮົາສົນໃຈ

ປະໂຫຍດຂອງການນຳ Fuzzy Logic ນຳມາປະຍຸກໃຊ້ງານ:

1. ຫຼັກການທີ່ທຳຄວາມເຂົ້າໃຈບໍ່ໄດ້ຍາກ.
2. ໃຊ້ຫຼັກການຄະນິດສາດໃນການສະຫຼຸບຄວາມ ເຮັດໃຫ້ມີຄວາມໃກ້ຄຽງກັບຄວາມເປັນຈິງ.
3. ເປັນຫຼັກການທີ່ມີຄວາມຢືດຢຸ່ນຕໍ່ຂໍ້ມູນ ເຮັດໃຫ້ມີຄວາມຖືກຕ້ອງແມ່ນຍຳໃນການຫາຜົນຮັບຫຼາຍຂຶ້ນ.
4. ແກ້ໄຂບັນຫາທີ່ບໍ່ມີຮູບແບບຈະແຈ້ງໄດ້ຢ່າງດີ ໂດຍບໍ່ຕ້ອງອາໄສການຕັດສິນໃຈຕາມກົດເກນສະເໝີໄປ.
5. ເປັນການປະສານງານລະຫວ່າປະສົດທິພາບຂອງເຄື່ອງຄອມພິວເຕີ ແລະ ສະໝອງຂອງຜູ້ຊ່ຽວຊານ.
6. ໃຊ້ພາສາທຳມະຊາດໃນການລະບຸຄ່າ ຫຼື ໃຊ້ຄວາມໝາຍພື້ນຖານຂອງການຕິດຕໍ່ສື່ສານລະຫວ່າງມະນຸດ

ບົດທີ 5 ແນະນຳການຮຽນຮູ້ຂອງເຄື່ອງຈັກ: (Introduction to Machine Learning)

ເຕັກໂນໂລຊີປັນຍາປະດິດມັກກ່ຽວຂ້ອງກັບການຄວບຄຸມເຄື່ອງຈັກ ຫຼື ເຄື່ອງຄອມພິວເຕີໃຫ້ສາມາດເຮັດວຽກໄດ້ຕາມທີ່ຕ້ອງການ, ມີປະສິດທິພາບຄືກັນກັບການຕັດສິນໃຈໂດຍມະນຸດ. ເຄື່ອງຈັກ ຫຼື ເຄື່ອງຄອມພິວເຕີເຫຼົ່ານັ້ນຈຳເປັນຕ້ອງມີການຮຽນຮູ້ສະຖານະການຕ່າງໆ ເພື່ອໃຫ້ເຂົ້າໃຈ ແລະ ຕັດສິນໃຈແກ້ໄຂບັນຫາຄ້າຍຄືກັບມະນຸດໄດ້ເອີ້ນຂະບວນການນີ້ວ່າ “ການຮຽນຮູ້ຂອງເຄື່ອງຈັກ (Machine Learning)” ເຊິ່ງເປັນເຕັກໂນໂລຊີທີ່ເນັ້ນໃສ່ການພັດທະນາ ແລະ ອອກແບບຂັ້ນຕອນວິທີທີ່ສະໜັບສະໜູນໃຫ້ເຄື່ອງຈັກຕັດສິນໃຈແທນມະນຸດໄດ້ສຳລັບບົດນີ້ຈະແນະນຳໃຫ້ຮູ້ຈັກກັບການຮຽນຮູ້ຂອງເຄື່ອງຈັກ, ວິທີການສ້າງເຕັກນິກການຮຽນຮູ້ ແລະ ການສ້າງການຕັດສິນໃຈແບບຕົ້ນໄມ້ (Decision Tree) ທີ່ສາມາດນຳໄປໃຊ້ໃນການສ້າງກົດ ຫຼື ຫຼັກການສຳລັບຄວາມຮູ້.

5.1 ການຮຽນຮູ້ຂອງເຄື່ອງຈັກ

ໃນປັນຍາປະດິດການຮຽນຮູ້ເປັນປັດໄຈສຳຄັນທີ່ເຮັດໃຫ້ເຄື່ອງຄອມພິວເຕີ ຫຼື ເຄື່ອງຈັກສາມາດຕັດສິນໃຈ, ເຂົ້າໃຈ ແລະ ແກ້ໄຂບັນຫາຕາມສະຖານະການທີ່ເກີດຂຶ້ນໄດ້. ໂດຍທົ່ວໄປເຄື່ອງຄອມພິວເຕີ ຫຼື ເຄື່ອງຈັກຈຳເປັນຕ້ອງອາໄສມະນຸດເປັນຜູ້ຄວບຄຸມການເຮັດວຽກທຸກຂັ້ນຕອນ ຕັ້ງແຕ່ປ້ອນຂໍ້ມູນຈົນເຖິງການສະແດງຜົນຮັບ. ດ້ວຍເຫດນີ້ຈຶ່ງມີແນວຄິດທີ່ຈະຫຼຸດບົດບາດການເຮັດວຽກຂອງມະນຸດລົງໂດຍໃຊ້ປັນຍາປະດິດທີ່ມີປະສິດທິພາບໃກ້ຄຽງກັບສະໝອງມະນຸດແທນ. ການທີ່ຈະໃຫ້ເຄື່ອງຈັກເຮັດວຽກໄດ້ຢ່າງໜ້າເຊື່ອຖື, ຖືກຕ້ອງ ແລະ ວ່ອງໄວໄດ້ນັ້ນ ປັນຍາປະດິດທີ່ຢູ່ໃນເຄື່ອງຈັກຕ້ອງມີການຮຽນຮູ້ຂໍ້ມູນຕ່າງໆ ເພື່ອໃຫ້ສາມາດປະຕິບັດວຽກໄປພ້ອມໆກັບການພັດທະນາປະສິດທິພາບຂອງວຽກໃຫ້ສູງຂຶ້ນ. ຄ້າຍຄືກັບການຮຽນຮູ້ຂອງມະນຸດທີ່ສາມາດຈື່ຈຳ ແລະ ສຶກສາວິທີແກ້ໄຂບັນຫາໃນສະຖານະການທີ່ຄ້າຍຄືກັນໄດ້, ເມື່ອມີປະສົບການຫຼາຍກໍ່ສາມາດແກ້ບັນຫາໄດ້ຢ່າງມີປະສິດທິພາບຫຼາຍຂຶ້ນ.

ເມື່ອເວົ້າເຖິງການຮຽນຮູ້ຂອງເຄື່ອງຈັກໃນດ້ານວິທະຍາສາດຄອມພິວເຕີ ຈະໝາຍເຖິງເຕັກນິກ ຫຼື ຂະບວນການທີ່ໃຊ້ປັບແຕ່ງ, ປັບປຸງອຸປະກອນເຄື່ອງຈັກ ຫຼື ຄອມພິວເຕີໃຫ້ມີຄວາມສາມາດເພີ່ມຂຶ້ນ ມີພຶດຕິກຳສະເພາະຕົວທີ່ສະໜັບສະໜູນໃຫ້ອຸປະກອນເຫຼົ່ານີ້ສາມາດຮຽນຮູ້, ເຂົ້າໃຈ ແລະ ສຶກສາຂໍ້ມູນຈາກປະສົບການ ຫຼື ການສະແດງຜົນທີ່ຜ່ານມາໃນອາດີດໄດ້, ເພື່ອໃຊ້ເປັນຂໍ້ມູນສຳລັບພັດທະນາການໃນການເຮັດວຽກຂັ້ນຕໍ່ໄປ. ຈາກທີ່ໄດ້ກ່າວມາເປັນຄວາມໝາຍລວມຂອງການຮຽນຮູ້ຂອງເຄື່ອງຈັກ, ເຊິ່ງເປັນວິທີການທີ່ຊ່ວຍໃຫ້ເຄື່ອງຈັກສາມາດຮຽນຮູ້ໄດ້ຄືມະນຸດ. ສຳລັບປັນຍາປະດິດຄຳວ່າ ການຮຽນຮູ້ຂອງເຄື່ອງຈັກຈະມີຄວາມໝາຍສະເພາະເຈາະຈົງລົງໄປເຖິງການນຳໃຊ້ເຕັກນິກຕ່າງໆ ມາພັດທະນາເຄື່ອງຈັກໃຫ້ມີປະສິດທິພາບໃນການຮຽນຮູ້ສິ່ງຕ່າງໆ ປຽບຄືການສ້າງສະໝອງໃຫ້ກັບເຄື່ອງຈັກເພື່ອໃຊ້ໃນການຮຽນຮູ້ ແລະ ແກ້ໄຂບັນຫາ.

ການຮຽນຮູ້ຂອງເຄື່ອງຈັກໃນປັນຍາປະດິດຈະກ່ຽວຂ້ອງກັບການພັດທະນາຂັ້ນຕອນວິທີ ຫຼື ເຕັກນິກຕ່າງໆ ທີ່ຊ່ວຍໃຫ້ເຄື່ອງຈັກ ຫຼື ເຄື່ອງຄອມພິວເຕີມີຄວາມສາມາດໃນການຮຽນຮູ້ຂໍ້ມູນທີ່ສົນໃຈໄດ້ ເພື່ອຈະນຳໄປໃຊ້ໃນການແກ້ໄຂບັນຫາສະເພາະໜ້າ ທີ່ຕ້ອງພົບພໍ້ຄືກັບການຮຽນຮູ້ຂອງມະນຸດທີ່ອາໄສປະສົບການທີ່ໄດ້ຈາກເຫດການຕ່າງໆທີ່ເຄີຍຜ່ານ ມາເກັບສະສົມໄວ້ເພື່ອເປັນຄວາມຮູ້ເຊິ່ງສາມາດນຳມາໃຊ້ໃນການແກ້ໄຂບັນຫາ ແລະ ຄິດຄົ້ນວິທີການທີ່ເໝາະສົມຈົນໄດ້ຜົນຮັບທີ່ຕ້ອງການ.

ການຮຽນຮູ້ໂດຍທົ່ວໄປແບ່ງໄດ້ 2 ຊະນິດດັ່ງນີ້:

1. Deductive

ເປັນການຮຽນຮູ້ໂດຍອາໄສຄວາມຮູ້ທີ່ເປັນຈິງຢູ່ແລ້ວ ມີຄວາມຈິງເປັນສາກົນ ເຊິ່ງທຸກຄົນໄດ້ຍອມຮັບ ໂດຍສາມາດຄາດການໄດ້ວ່າເຫດການດັ່ງກ່າວຈະເກີດຂຶ້ນແນ່ນອນຕາມຮູບແບບຂອງສິ່ງແວດລ້ອມ, ເຊັ່ນ: ໃນກັບໜ່ວຍໜຶ່ງບັນຈຸໝາກບານຢູ່ 2 ໜ່ວຍ ຄື: ໜ່ວຍສີຟ້າ ແລະ ສີແດງ, ສະແດງວ່າໂອກາດທີ່ຈະຈັບໄດ້ໝາກບານ 1 ໜ່ວຍ ເປັນສີໃດສີໜຶ່ງ ຖືວ່າໄດ້ 1 ໃນ 2 ໜ່ວຍ ເທົ່າກັບ 50%, ເຊິ່ງເປັນຄ່າກະຕວງທີ່ທຸກຄົນຍອມຮັບກັນ ແລະ ເຊື່ອວ່າຫາກຈັບໝາກບານ 1 ໜ່ວຍ ໂອກາດທີ່ຈະໄດ້ໝາກບານສີໃດໆ ຈະເທົ່າກັນເປັນຕົ້ນ.

2. Inductive

ເປັນການຮຽນຮູ້ຈາກເຫດການ ຫຼື ສິ່ງທີ່ສົນໃຈ ໂດຍຮູ້ຂໍ້ມູນ ຫຼື ຄ່າຄວາມຈິງພຽງບາງສ່ວນ, ເຊິ່ງຈະເອົາມາໃຊ້ເປັນຂໍ້ມູນໃນການສຶກສາ ແລະ ສ້າງຄວາມເຂົ້າໃຈ ເພື່ອໃຫ້ໄດ້ຄວາມຈິງຂອງຂໍ້ມູນສ່ວນອື່ນໆ ທັງໝົດ ຈົນເປັນຄວາມຈິງສາກົນທີ່ຄົນສ່ວນຫຼາຍຍອມຮັບ ເຊັ່ນ: ການຮຽນຮູ້ຂອງພະນັກງານຂາຍ ໂດຍສຶກສາຈາກພຶດຕິກຳ, ລັກສະນະຂອງລູກຄ້າ ແລະ ຄວາມສົນໃຈໃນຂະນະນຳສະເໜີສິນຄ້າ ເພື່ອຄົ້ນຫາຄວາມຕ້ອງການທີ່ແທ້ຈິງຂອງລູກຄ້າ, ຊ່ວຍໃຫ້ຮູ້ເຖິງແນວທາງໃນການຂາຍສິນຄ້າແກ່ລູກຄ້າໃຫ້ປະສົບຜົນສຳເລັດ. ເຊິ່ງເປັນການສຶກສາຈາກຂໍ້ມູນພຽງບາງສ່ວນ ຈົນຮູ້ເຖິງຄວາມຕ້ອງການທີ່ແທ້ຈິງຂອງລູກຄ້າ ເປັນຕົ້ນ.

ສຳລັບການຮຽນຮູ້ຂອງເຄື່ອງຈັກຈະເປັນການຮຽນຮູ້ຈາກຂໍ້ມູນເປັນບາງສ່ວນ ກ່ອນຈະດຳເນີນການຫາຄວາມຈິງທີ່ເປັນສາກົນ. ໃນຂະບວນການຂອງການຮຽນຮູ້ຂອງເຄື່ອງຈັກຈະມີການນຳເຕັກນິກ ຫຼື ຂັ້ນຕອນວິທີທີ່ມີຮູບແບບແຕກຕ່າງກັນມາແກ້ບັນຫາໃນລັກສະນະຕ່າງໆ, ເຊິ່ງຂັ້ນຕອນວິທີໃນການຮຽນຮູ້ຂອງເຄື່ອງຈັກມີຢູ່ຫຼາຍຊະນິດ ສຳລັບໃນບົດນີ້ຈະກ່າວເຖິງຂັ້ນຕອນວິທີບາງຊະນິດ ໄດ້ແກ່ Supervised Learning ແລະ Unsupervised Learning.

5.1.1 Supervised Learning

ເປັນການຮຽນຮູ້ທີ່ສາມາດນຳສະເໜີ ແລະ ຈຳແນກຂໍ້ມູນພາຍໃນຊຸດຂໍ້ມູນວ່າມີຜົນຮັບທີ່ຖືກ ຫຼື ຜິດໄດ້, ເຊິ່ງໃນຊຸດຂໍ້ມູນດັ່ງກ່າວຈະປະກອບດ້ວຍຂໍ້ມູນເອກະລາດ ແລະ ຂໍ້ມູນທີ່ສົນໃຈ. ຂໍ້ມູນເຫຼົ່ານີ້ຈະຖືກເອົາໄປໃຊ້ປະມານຄ່າ ຫຼື ພະຍາກອນຄ່າຂໍ້ມູນ, ໂດຍມີພື້ນຖານການພະຍາກອນຈາກຂໍ້ມູນທັງໝົດໃນຊຸດຂໍ້ມູນ. ຕົວຢ່າງເຕັກນິກທີ່ໃຊ້ໃນການຮຽນຮູ້ປະເພດນີ້ຄື: Decision Tree, Perceptrons ແລະ Backpropagation ເປັນຕົ້ນ.

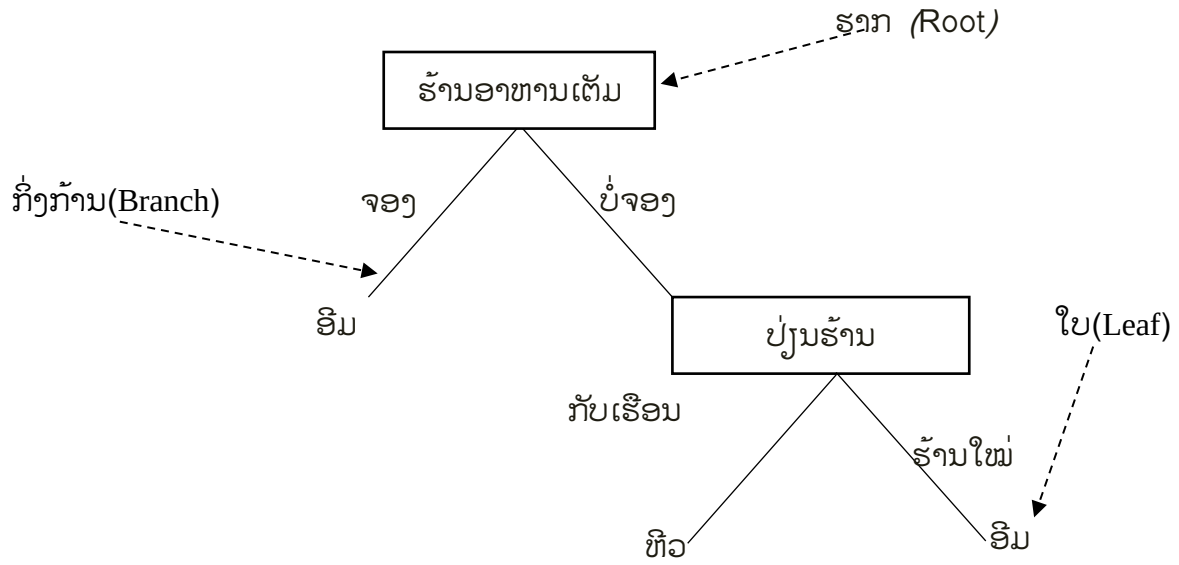
5.1.2 Unsupervised Learning

ເປັນການຮຽນຮູ້ທີ່ບໍ່ມີການກຳນົດຂໍ້ມູນທີ່ສົນໃຈພາຍໃນຊຸດຂໍ້ມູນ, ຈຶ່ງບໍ່ມີການຈຳແນກຂໍ້ມູນວ່າມີຜົນຮັບເປັນແນວໃດ, ແຕ່ຈະເປັນການຮຽນຮູ້ທີ່ກ່ຽວຂ້ອງກັບຄວາມສຳພັນຂອງຂໍ້ມູນ ເຊິ່ງຈະນຳໄປໃຊ້ໃນການຈຳແນກ ແລະ ແຍກຂໍ້ມູນອອກເປັນກຸ່ມ. ຕົວຢ່າງເຕັກນິກທີ່ໃຊ້ໃນການຮຽນຮູ້ປະເພດນີ້ຄື: Nearest Neighbor Classification ເປັນຕົ້ນ.

5.2 ການຕັດສິນໃຈແບບຕົ້ນໄມ້ (Decision Tree)

Decision Tree ຫຼື Classification Tree ເປັນອີກເຕັກນິກໜຶ່ງຂອງການຮຽນຮູ້ຂອງເຄື່ອງຈັກທີ່ໃຊ້ໃນການພັດທະນາການຮຽນຮູ້ຂອງເຄື່ອງຈັກ ຫຼື ເຄື່ອງຄອມພິວເຕີ. ເຕັກນິກນີ້ຈັດເປັນ Supervised Learning, ເຊິ່ງເປັນແບບຈຳລອງທີ່ໃຊ້ສຳລັບຄາດຄະເນ ຫຼື ທຳນາຍເຫດການທີ່ຈະເກີດຂຶ້ນລວງໜ້າ, ເຊິ່ງເປັນຜົນໄດ້ຮັບທີ່ໄດ້ຈາກການຕັດສິນໃຈ. Decision Tree ເປັນຂັ້ນຕອນໃນການຮຽນຮູ້ ທີ່ບໍ່ຄ່ອຍຊັບຊ້ອນປານໃດ ໂດຍຈະມີການແຕກແໜ່ງຈາກໂນດຮາກ (Root) ສູ່ໃບ (Leaf) ແລະ ມີກິ່ງກ້ານ (Branch) ແຕກອອກໄປຕາມເງື່ອນໄຂ ຫຼື ຂໍ້ມູນທີ່ໄດ້ຄາດຄະເນໄວ້ວ່າຈະເກີດຂຶ້ນ ເພື່ອໃຫ້ຮູ້ເຖິງຜົນຮັບຂອງແຕ່ລະເຫດການ, ເປັນແບບຈຳລອງທີ່ມີການເຊື່ອມຕໍ່ລະຫວ່າງສິ່ງທີ່ສົນໃຈກັບຜົນສະຫຼຸບທີ່ອາດຈະເກີດຂຶ້ນ ຈາກຄ່າຂອງເຫດການຕ່າງໆເຊັ່ນ: ເຈົ້າຫິວເຂົ້າບໍ່? ຄຳຕອບເປັນໄດ້ທັງຫິວ ຫຼື ບໍ່ຫິວເປັນຕົ້ນ.

ໂຄງສ້າງຂອງ Decision Tree ຈະປະກອບດ້ວຍໃບເປັນສ່ວນຂອງຂໍ້ມູນທີ່ເຮົາສົນໃຈ, ເຊິ່ງອາດເປັນຂໍ້ມູນທີ່ເກີດຂຶ້ນໂດຍສະພາບແວດລ້ອມຕາມສະຖານະການນັ້ນ ຫຼື ເປັນສິ່ງທີ່ກຳໜົດຕາມການຄາດຄະເນວ່າມີໂອກາດທີ່ຈະເກີດຂຶ້ນຕາມເຫດການສະພາບແວດລ້ອມ, ໂດຍແຕ່ລະໃບຈະຖືກເຊື່ອມດ້ວຍກິ່ງກ້ານ (Branch) ເຊິ່ງເປັນຂໍ້ມູນທີ່ແຕກອອກມາຈາກໂນດຕ່າງໆ ປຽບຄືເປັນກິ່ງກ້ານໃນການຕັດສິນໃຈວ່າຈະໃຫ້ເກີດເຫດການໃດຂຶ້ນ ມີກຳເນີດຈາກໂນດເທິງສຸດແມ່ນ ຮາກ (Root) ເຊິ່ງຈະສົ່ງມີຜົນຕໍ່ຜົນໄດ້ຮັບທີ່ແຕກຕ່າງກັນ, ເຊັ່ນ: ຮ້ານອາຫານບ່ອນນັ່ງເຕັມ ເຮົາຈະຈອງ ຫຼືບໍ? ຖ້າຕັດສິນໃຈວ່າບໍ່ຈອງ ເຫດການທີ່ເກີດຂຶ້ນກໍ່ຈະແຕກຕ່າງກັນໄປ ເຊັ່ນ: ເດີນທາງກັບບ້ານ ຫຼື ເດີນທາງໄປຫາຮ້ານອາຫານໃໝ່ເປັນຕົ້ນ. ຕົວຢ່າງແບບຈຳລອງ Decision Tree ສະແດງດັ່ງຮູບທີ 5.1

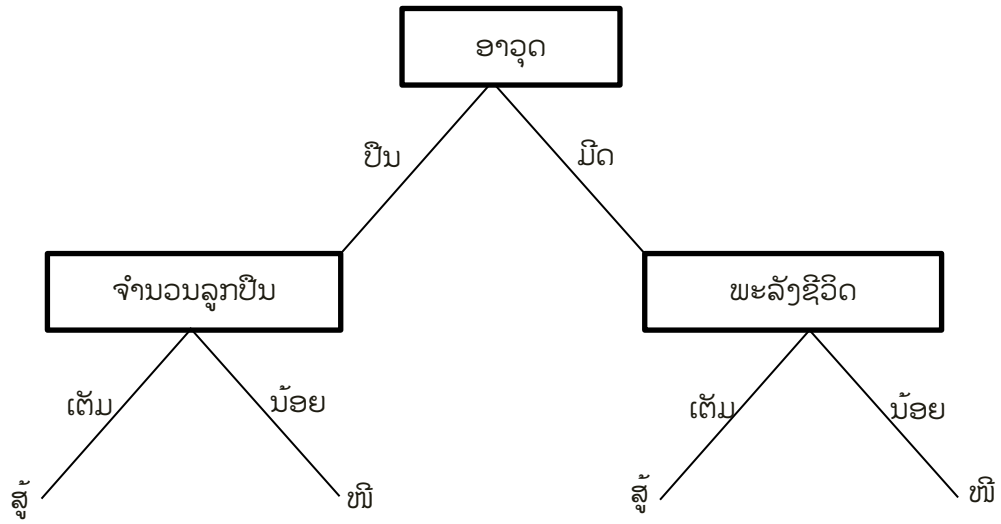


ຮູບທີ 5.1 ການສະແດງເຖິງເຫດການຮ້ານອາຫານເຕັມແລ້ວ

ຈາກຮູບທີ 5.1 ເປັນການສະແດງເຖິງເຫດການຮ້ານອາຫານເຕັມແລ້ວ ຈະຕັດສິນໃຈແນວໃດ, ເຊິ່ງເລືອກໄດ້ວ່າຈະປ່ຽນຮ້ານ ຫຼື ວ່າຈະຈອງໄວ້. ໃນນີ້ມີປັດໃຈເລື່ອງເວລາເຂົ້າມາກ່ຽວຂ້ອງເຮັດໃຫ້ສາມາດຕັດສິນໃຈໄດ້ວ່າຈະຈອງໂຕະ ຫຼືບໍ່? ຈາກທີ່ກ່າວມາຈະໄຈ້ແຍກສິ່ງທີ່ສົນໃຈໄດ້ 3 ຢ່າງຄື: ຈຳນວນລູກຄ້າ, ໄລຍະເວລາໃນການລໍຖ້າ ແລະ ການຈອງໂຕະ. ນອກຈາກເຫດການເຫຼົ່ານີ້ແລ້ວ ອາດມີເຫດການເພີ່ມເຕີມອື່ນທີ່ສາມາດຄາດຄະເນວ່າຈະເກີດໄດ້ອີກ ທັງນີ້ຂຶ້ນຢູ່ກັບຄວາມຕ້ອງການ ຫຼື ການຄາດຫວັງວ່າຜົນຮັບຈະອອກມາຄືແນວໃດ.

5.3. ການຮຽນຮູ້ດ້ວຍ Decision Tree

ການຮຽນຮູ້ດ້ວຍ Decision Tree ເປັນການຮຽນຮູ້ຈາກການຄາດຄະເນເຫດການຕ່າງໆທີ່ອາດເກີດຂຶ້ນ, ເຊິ່ງຈະອາໄສເງື່ອນໄຂເປັນຕົວຊ່ວຍການຕັດສິນໃຈວ່າເມື່ອເກີດເຫດການໃດໜຶ່ງຂຶ້ນ ຈະແດງອອກມາແນວໃດ. ໃນການຄາດຄະເນຈະຖືກນຳສະເໜີດ້ວຍຮູບແບບການຕັດສິນໃຈທີ່ມີເງື່ອນໄຂເປັນ ”ຖ້າ...ແລ້ວ” (if/then/else) ເຮັດໃຫ້ສາມາດຮຽນຮູ້ໄດ້ຕາມເງື່ອນໄຂທີ່ຈະເກີດຂຶ້ນໃນສະຖານະການທີ່ແຕກຕ່າງກັນເຊັ່ນ: ການຮຽນຮູ້ໃນການຫຼີ້ນເກມທີ່ຜູ້ຫຼີ້ນຈະມີພຶດຕິກຳຄ້າຍຄືກັນເມື່ອຢູ່ໃນສະຖານະການທີ່ມີຂອບເຂດ ແລະ ຕົວປ່ຽນຈຳກັດ, ໃນກໍລະນີຜູ້ຫຼີ້ນທີ່ໃຊ້ອາວຸດປືນສາມາດຍິງໃນໄລຍະໄກໄດ້, ກ້າທີ່ຈະຕໍ່ສູ້ກັບສັດຕູເຖິງແມ່ນວ່າພະລັງຊີວິດຈະເຫຼືອໜ້ອຍກໍຕາມ, ແຕ່ຜູ້ຫຼີ້ນທີ່ໃຊ້ອາວຸດມີດທີ່ຕ້ອງຕໍ່ສູ້ໃນໄລຍະໃກ້ຈຳເປັນຕ້ອງຄຳນຶງເຖິງປັດໄຈດ້ານພະລັງຊີວິດຈະຕ້ອງເຫຼືອຢູ່ໃຫ້ຫຼາຍ ຫຼື ສັດຕູຕ້ອງບໍ່ເກ່ງປານໃດ, ໃນນີ້ອາດຂຶ້ນຢູ່ກັບສະພາບແວດລ້ອມໃນຕອນນັ້ນ. ຕົວຢ່າງ Decision Tree ແບບງ່າຍສະແດງດັ່ງຮູບ 5.2



ຮູບທີ 5.2 ສະແດງຕົວຢ່າງຂອງ Decision Tree ແບບງ່າຍ

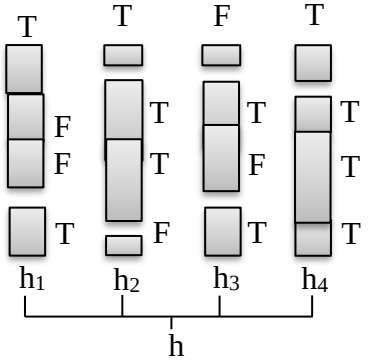
ຮູບທີ 5.2 ຜູ້ຫຼີ້ນທີ່ໃຊ້ອາວຸດຕ່າງກັນຈະມີເງື່ອນໄຂໃນການຕັດສິນໃຈທີ່ແຕກຕ່າງກັນ ໂດຍຜູ້ຫຼີ້ນທີ່ໃຊ້ປືນຈະຕ້ອງຄຳໜຶ່ງເຖິງຈຳນວນລູກປືນທີ່ໃຊ້ຍິງໃສ່ສັດຕູວ່າສາມາດເອົາຊະນະສັດຕູໄດ້ ຫຼືບໍ່? ໃນຂະນະທີ່ຜູ້ຫຼີ້ນທີ່ໃຊ້ມີດຈຳເປັນຕ້ອງພິຈາລະນາວ່າພະລັງຊີວິດຂອງຕົນເອງເຫຼືອພໍທີ່ຈະຕໍ່ສູ້ກັບສັດຕູ ແລະ ເອົາຊະນະໄດ້ ຫຼືບໍ່? ຫາກທັງສອງເງື່ອນໄຂມີບໍ່ພຽງພໍຜູ້ຫຼີ້ນຈະຕັດສິນໃຈຫຼົບໜີຫຼາຍກວ່າທີ່ຈະປະເສີນໜ້າຕໍ່ສູ້.

5.4. Ensemble Learning.

ການຮຽນຮູ້ທີ່ກ່າວມາກ່ອນໜ້ານີ້ເປັນການຮຽນຮູ້ ແລະ ການຄາດຄະເນທີ່ມີຂໍ້ມູນສະພາບແວດລ້ອມມາຈາກສົມມຸຕິຖານດຽວເທົ່ານັ້ນ ຫຼື ທີ່ເອີ້ນວ່າ “Single Hypothesis”, ແຕ່ປັນຍາປະດິດຈະຕ້ອງຮຽນຮູ້ ແລະ ວິເຄາະຂໍ້ມູນທີ່ໄດ້ມາຈາກຫຼາຍສົມມຸດຕິຖານເພື່ອການຕັດສິນໃຈ. ໝາຍຄວາມວ່າຂໍ້ມູນໜຶ່ງຊຸດອາດສ້າງເປັນ Decision Tree ທີ່ມີກິ່ງກ້ານ ແລະ ໂໝດຈຳນວນຫຼາຍ ຈຶ່ງຈຳເປັນຕ້ອງມີການຈຳແນກ, ແຍກຍ້າຍ, ຈັດກຸ່ມລວມເຖິງຄັດເລືອກຕົວຢ່າງທີ່ດີທີ່ສຸດ ສຳລັບເອົາໄປໃຊ້ວຽກຕໍ່ໄປຕາມຂະບວນການ ຫຼື ຂັ້ນຕອນການຮຽນຮູ້. ດ້ວຍເຫດນີ້ຈຶ່ງມີທິດສະດີທີ່ຊ່ວຍສະໜັບສະໜູນໃຫ້ປັນຍາປະດິດທີ່ພັດທະນາຂຶ້ນສາມາດລວບລວມ ແລະ ເລືອກທີ່ຈະວິເຄາະສົມມຸດຕິຖານ ເພື່ອຄັດເລືອກຂໍ້ມູນທີ່ເໝາະສົມ ແລະ ມີປະສິດທິພາບໄດ້. ທິດສະດີນີ້ເອີ້ນວ່າ: “Ensemble Learning” [Russell and Norvig, 2003].

ສົມມຸດຕິຖານໃນ Ensemble Learning ຈະຖືກຈັດກຸ່ມ ແລະ ລວບລວມກ່ອນທີ່ຈະເອົາໄປໃຊ້ວິເຄາະ ເນື່ອງຈະຕ້ອງເຮັດການຄັດເລືອກກຸ່ມ ຫຼື ຊຸດຂອງຂໍ້ມູນຈາກສົມມຸດຕິຖານຕ່າງໆ ທີ່ມີຄວາມເປັນໄປໄດ້ ແລະ ເໝາະສົມທີ່ສຸດໃນການນຳໄປວິເຄາະ ເພື່ອຫາຂໍ້ມູນທີ່ດີທີ່ສຸດຕໍ່ໄປ. ໂດຍແຕ່ລະສົມມຸດຕິຖານຈະມີສ່ວນທີ່ສຳພັນກັນ ແລະ ສາມາດເອົາໄປໃຊ້ເປັນຂໍ້ມູນ ຫຼື ແນວທາງໃນການຮຽນຮູ້, ເຊິ່ງເປັນສ່ວນທີ່ກ່ຽວຂ້ອງກັບສິ່ງທີ່ສົນໃຈຢ່າງແທ້ຈິງ. ຫຼັກການຂອງ Ensemble Learning ນີ້ຊ່ວຍໃຫ້ການຮຽນຮູ້ມີປະສິດຕິພາບສູງຂຶ້ນ ລວມເຖິງສາມາດແກ້ໄຂບັນຫາທີ່ມີຄວາມຊັບຊ້ອນຫຼາຍໄດ້.

ສຳລັບວິທີການທີ່ນິຍົມໃຊ້ທິດສະດີ Ensemble Learning ມາໃຊ້ຄື: “Boosting” ເຊິ່ງເປັນຫຼັກການທີ່ອາໄສການກຳນົດນ້ຳໜັກໃຫ້ກັບຊຸດຂໍ້ມູນ ຫຼື ເອີ້ນວ່າ: “Weighted Training Set” ໂດຍໃນແຕ່ລະຊຸດຂໍ້ມູນຈະມີຄ່ານ້ຳໜັກກຳນົດໃຫ້ຫຼາຍກ່ວາ ຫຼື ເທົ່າກັບ 0, ຄ່ານ້ຳໜັກຂອງຊຸດຂໍ້ມູນໃດສູງກ່ວາສະແດງວ່າມີຄວາມສຳຄັນທີ່ຈະຕ້ອງພິຈາລະນາເປັນອັນດັບທຳອິດໃນຄວາມຮູ້, ອີກຢ່າງໜຶ່ງຊຸດຂໍ້ມູນດັ່ງກ່າວຍັງບົ່ງບອກຄວາມສຳຄັນຂອງສົມມຸດຕິຖານທີ່ກ່ຽວຂ້ອງກັບຊຸດຂໍ້ມູນນັ້ນໄດ້ອີກ. ຫຼັກການ Boosting ຈະເລີ່ມດ້ວຍການກຳນົດຄ່ານ້ຳໜັກໃຫ້ຊຸດຂໍ້ມູນເທົ່າກັບ 1 ທັງໝົດ, ເຊິ່ງຄ່າດັ່ງກ່າວຈະເຮັດໃຫ້ເກີດຂໍ້ມູນສົມມຸດຕິຖານ h_1 ຂຶ້ນ ໂດຍຊຸດຂໍ້ມູນນີ້ເມື່ອເຮັດການວິເຄາະແລ້ວຈະມີທັງຂໍ້ມູນທີ່ຖືກ ແລະ ຜິດປົນກັນ ເຮັດໃຫ້ຫຼັງການວິເຄາະ h_1 ຄ່ານ້ຳໜັກຂອງຊຸດຂໍ້ມູນຈະປ່ຽນແປງໄປ ມີທັງເພີ່ມຂຶ້ນ ແລະ ລຸດລົງເພື່ອໃຫ້ສາມາດຈັດກຸ່ມຊຸດໄດ້ຖືກຕ້ອງ ແລະ ເໝາະສົມ, ເຊິ່ງຈະກຳນົດການວິເຄາະຕໍ່ໄປໃນຂັ້ນຕອນຂອງ h_2 ແລະ ຂັ້ນຕອນຕໍ່ໄປຈົນກ່ວາຈະໄດ້ຕາມທີ່ຕ້ອງການ ດັ່ງຮູບທີ 5.3.



ຮູບທີ 5.3 ສະແດງຂັ້ນຕອນການ Boosting ທີ່ໃຊ້ທິດສະດີ Ensemble Learning

ຮູບທີ 5.3 ເປັນການດຳເນີນການດ້ວຍຫຼັກການ Boosting ເຊິ່ງຈະເລີ່ມຈາກການກຳນົດຄ່ານ້ຳໜັກໃຫ້ກັບຊຸດຂໍ້ມູນໃນ h_1 ເທົ່າກັນທັງໝົດ ແລະ ດຳເນີນການຕໍ່ໄປດ້ວຍການປ່ຽນແປງຄ່ານ້ຳໜັກຂອງແຕ່ລະຊຸດຂໍ້ມູນຈົນກ່ວາຈະໄດ້ຊຸດຂໍ້ມູນທີ່ມີປະສິດຕິພາບ ແລະ ເໝາະສົມທີ່ສຸດ. ໃນການປັບຄ່ານ້ຳໜັກສາມາດເຮັດໄດ້ຫຼາຍຮູບແບບທັງຫຼຸດຄ່ານ້ຳໜັກ ແລະ ເພີ່ມຄ່ານ້ຳໜັກ. ປັດໄຈເຫຼົ່ານີ້ຊ່ວຍໃຫ້ຊຸດຂໍ້ມູນມີຄ່ານ້ຳໜັກທີ່ເໝາະສົມທີ່ສຸດ, ທິດສະດີການລວມຊຸດຂໍ້ມູນ ຫຼື ສົມມຸດຕິຖານເຂົ້າກັນຕາມຫຼັກການຂອງ Ensemble Learning ຊ່ວຍໃຫ້ການຄາດຄະເນ ແລະ ການວິເຄາະຂໍ້ມູນມີປະສິດທິພາບສູງຂຶ້ນ, ເນື່ອງຈາກມີການປັບປ່ຽນນ້ຳໜັກຂອງຊຸດຂໍ້ມູນເພື່ອໃຫ້ມີຄວາມເໝາະສົມຢູ່ສະເໝີ ເຮັດໃຫ້ຂໍ້ຜິດພາດ ຫຼື ຊຸດຂໍ້ມູນທີ່ສົ່ງຜົນໃຫ້ເກີດຄວາມຜິດພາດມີນ້ອຍລົງ. ດັ່ງນັ້ນ, ຈຶ່ງຊ່ວຍໃຫ້ການພັດທະນາລະບົບການຮຽນຮູ້ສະແດງຜົນ ແລະ ສາມາດແກ້ໄຂ ພ້ອມທັງຄົ້ນຫາແນວທາງທີ່ເໝາະສົມກັບບັນຫາໄດ້ດີຂຶ້ນ.

5.5. ຕົວຢ່າງການປະຍຸກໃຊ້ Decision Tree.

ໃນການສ້າງ Decision Tree ອາໄສຄ່າຄວາມນ່າຈະເປັນ ທີ່ຈຳເປັນຕ້ອງຄຳນວນຫາຈາກຂໍ້ມູນດິບທີ່ໄດ້ມາ, ເຊິ່ງອາດໄດ້ມາຈາກການສຳຫຼວດ ຫຼື ມີການກຳນົດຂຶ້ນຈາກແຫຼ່ງຂໍ້ມູນໂດຍກົງ. ການຫາຄ່າຄວາມໜ້າຈະເປັນຈະພິຈາລະນາຈາກຂໍ້ມູນທີ່ຢູ່ໃນຊຸດດຽວກັນທັງໝົດເພື່ອຄຳນວນຫາຄ່າຂອງ Gain Function ຕໍ່ໄປ, ໂດຍຈະເອົາມາປຽບທຽບກັນວ່າຊຸດຂໍ້ມູນໃດເໝາະສົມທີ່ຈະເປັນຮາກຂອງ Decision

Tree ຫຼາຍທີ່ສຸດ. ເຊິ່ງໄດ້ເວົ້າໄປແລ້ວໃນຫົວຂໍ້ທີ່ຜ່ານມາ, ສຳລັບຫົວຂໍ້ນີ້ຈະເວົ້າສະເພາະການປະຍຸກໃຊ້ Decision Tree ເນື່ອງຈາກເປັນທີ່ນິຍົມນຳໃຊ້ກັນຢ່າງຫຼວງຫຼາຍ, ໂດຍຫົວຂໍ້ນີ້ຈະເປັນຕົວຢ່າງໃນການສ້າງ Decision Tree ແລະ ການແປງເປັນກົດ ໂດຍໃຊ້ຂໍ້ມູນດັ່ງຕໍ່ໄປນີ້:

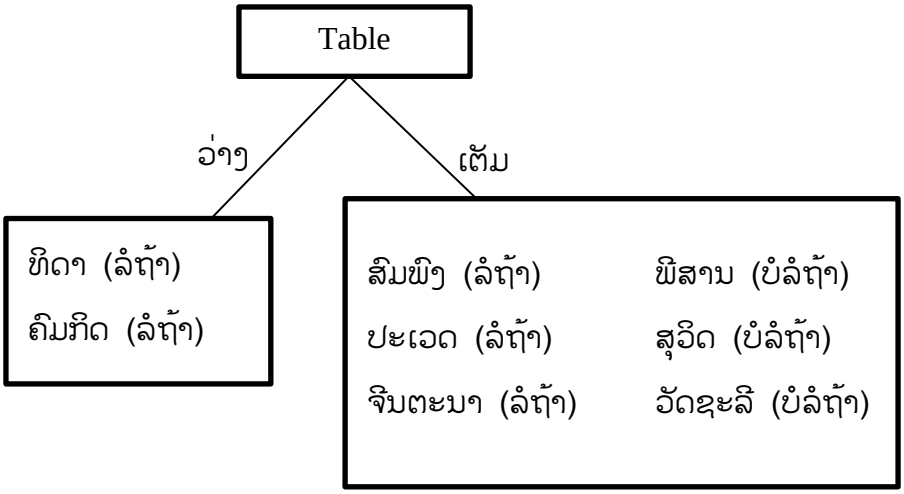
ມີການສຳຫຼວດລູກຄ້າທີ່ໃຊ້ບໍລິການຮ້ານອາຫານໃນສະຖານທີ່ແຫ່ງໜຶ່ງ ເມື່ອພົບກັບສະຖານະການທີ່ຮ້ານອາຫານທີ່ຕົນສົນໃຈເຂົ້າໃຊ້ບໍລິການບ່ອນນັ່ງເຕັມ ລູກຄ້າຈະມີພຶດຕິກຳທີ່ແຕກຕ່າງກັນ ເຊັ່ນ: ຫາກຮ້ານເຕັມຈະພິຈະລະນາວ່າສາມາດຈອງໂຕະໄດ້ ຫຼືບໍ່? ແລະ ໄລຍະເວລາທີ່ລໍຖ້າດົນປານໃດ ຫຼືອາດຈະພິຈາລະນາຮ້ານດັ່ງກ່າວເປັນຮ້ານອາຫານປະເພດໃດ, ລາຄາແພງຄຸ້ມກັບອາຫານ ຫຼືບໍ່? ເປັນຕົ້ນ. ໂດຍໃຫ້ຄວາມສົນໃຈກັບພຶດຕິກຳການລໍຖ້າທີ່ຈະໃຊ້ບໍລິການໃນຮ້ານອາຫານວ່າ ປັດໄຈທີ່ແຕກຕ່າງກັນຈະສົ່ງຜົນໃຫ້ລູກຄ້າອົດທົນລໍຖ້າໃຊ້ບໍລິການ ຫຼືບໍ່?. ສຳລັບຂໍ້ມູນຈາກລູກຄ້າທັງໝົດ 8 ຄົນ ເຊິ່ງມີຂໍ້ມູນດິບສະແດງໃນຕາຕະລາງທີ 5.2

ລູກຄ້າ	ສະຖານະຂອງໂຕະ	ລາຄາ	ຮ້ານອາຫານ	ໄລຍະເວລາລໍຖ້າ(ນາທີ)	ລູກຄ້າລໍຖ້າ
ສົມພົງ	ເຕັມ	ແພງ	ຍີ່ປຸ່ນ	10-30	ລໍຖ້າ
ຈັນຕະນາ	ເຕັມ	ຖືກ	ໄທ	10-30	ລໍຖ້າ
ພັສານ	ເຕັມ	ແພງ	ໄທ	10-30	ບໍ່ລໍຖ້າ
ສຸວິດ	ເຕັມ	ແພງ	ຍີ່ປຸ່ນ	ຫຼາຍກວ່າ 30	ບໍ່ລໍຖ້າ
ປະເວດ	ເຕັມ	ແພງ	ອິຕາລີ	0-10	ລໍຖ້າ
ທິດາ	ວ່າງ	ຖືກ	ໄທ	10-30	ລໍຖ້າ
ວັດຊະລີ	ເຕັມ	ແພງ	ອິຕາລີ	10-30	ບໍ່ລໍຖ້າ
ຄົມກິດ	ວ່າງ	ແພງ	ໄທ	0-10	ລໍຖ້າ

ຕາຕະລາງທີ 5.2 ສະແດງຂໍ້ມູນດິບຂອງການສຳຫຼວດຄວາມອົດທົນຂອງລູກຄ້າທີ່ລໍຖ້າໃຊ້ບໍລິການ.

ຈາກຕາຕະລາງ 5.2 ເຮັດໃຫ້ເຮົາຮູ້ວ່າລູກຄ້າແຕ່ລະຄົນມີຄວາມຕ້ອງການ ແລະ ພຶດຕິກຳໃນສະຖານະການໜຶ່ງໆ ທີ່ແຕກຕ່າງກັນ, ໂດຍຈະໃຫ້ຄວາມສົນໃຈກັບພຶດຕິກຳການລໍຖ້າຂອງລູກຄ້າວ່າຈະສາມາດລໍຖ້າເຂົ້າໃຊ້ບໍລິການຮ້ານຄ້ານັ້ນໄດ້ ຫຼືບໍ່? ໃນປັດໄຈທີ່ແຕກຕ່າງກັນ, ເຊິ່ງປັດໄຈດັ່ງກ່າວເປັນຊຸດຂໍ້ມູນທີ່ຈຳເປັນຕ້ອງພິຈາລະນາໃນການຄຳນວນຫາຄ່າ Gain Function ໄດ້ແກ່ສະຖານະຂອງໂຕະ (Table), ລາຄາອາຫານ (Price), ປະເພດຮ້ານອາຫານ (Type) ແລະ ໄລຍະເວລາໃນການລໍຖ້າ (Time) ສະແດງເປັນ Decision Tree ໄດ້ດັ່ງນີ້.

ສະຖານະຂອງໂຕະ (Table)



ຮູບທີ 5.4 ແບບຈຳລອງສະຖານະຂອງໂຕະ

ຈາກແບບຈຳລອງສະຖານະຂອງໂຕະ ເມື່ອແທນໃນ Gain Function ຈະໄດ້ດັ່ງນີ້:

$$\mathrm{E(s)} = -\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}$$

ດັ່ງນັ້ນຈະໄດ້

$$\mathrm{Gian(price)} = \left(-\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}\right) - \left(\frac{2}{8}\left(-\frac{2}{2}log_2\frac{3}{2}\right) + \frac{6}{8}\left(-\frac{3}{6}log_2\frac{3}{6} - \frac{3}{6}log_2\frac{3}{6}\right)\right)$$

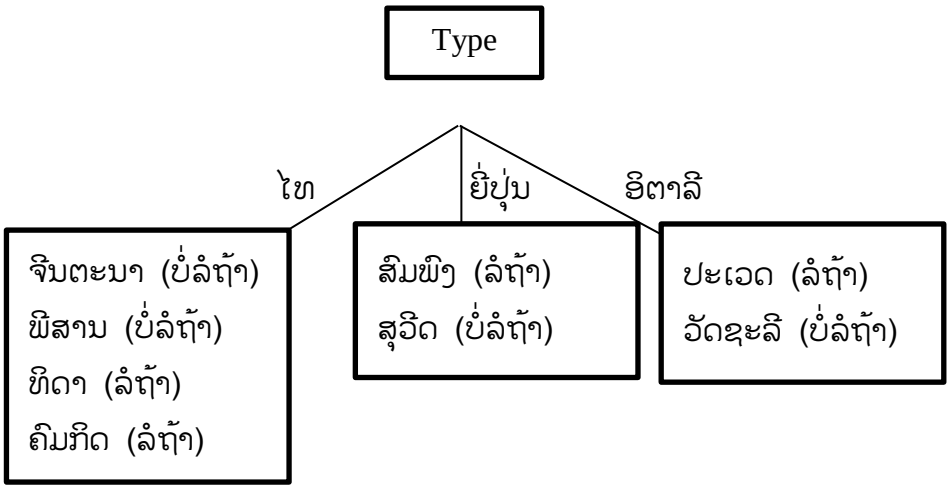
$$\mathrm{Gian(price)} = \left(-\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}\right) - \left(\frac{1}{4}(-log_2 1) + \frac{3}{4}\left(-\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2}\right)\right)$$

$$\mathrm{Gian(price)} = (0.4237 + 0.5306) - \left(-\frac{1}{4}(0) + \frac{3}{4}(0.5 + 0.5)\right)$$

$$\mathrm{Gian(price)} = (0.9543) - \left(\frac{3}{4}(1)\right)$$

$$\mathrm{Gian(price)} = 0.9543 - 0.75 = 0.2043$$

ປະເພດຮ້ານອາຫານ (Type)



ຮູບທີ 5.5 ແບບຈຳລອງປະເພດຮ້ານອາຫານ

ຈາກແບບຈຳລອງປະເພດຮ້ານອາຫານ ເມື່ອແທນໃນ Gain Function ຈະໄດ້ດັ່ງນີ້:

$$\mathrm{E(s)} = -\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}$$

ດັ່ງນັ້ນຈະໄດ້

$$\begin{aligned}\mathrm{Gain(Type)} = &\left(-\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}\right)\\ &-\left(\frac{4}{8}\left(-\frac{3}{4}log_2\frac{3}{4} - -\frac{1}{4}log_2\frac{1}{4}\right) + \frac{2}{8}\left(-\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2}\right)\right.\\ &\left.+\frac{2}{8}\left(-\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2}\right)\right)\end{aligned}$$

$$\begin{aligned}\mathrm{Gain(Type)} = &\left(-\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}\right)\\ &-\left(\frac{1}{2}\left(-\frac{3}{4}log_2\frac{3}{4} - -\frac{1}{4}log_2\frac{1}{4}\right) + \frac{1}{4}\left(-\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2}\right)\right.\\ &\left.+\frac{1}{4}\left(-\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2}\right)\right)\end{aligned}$$

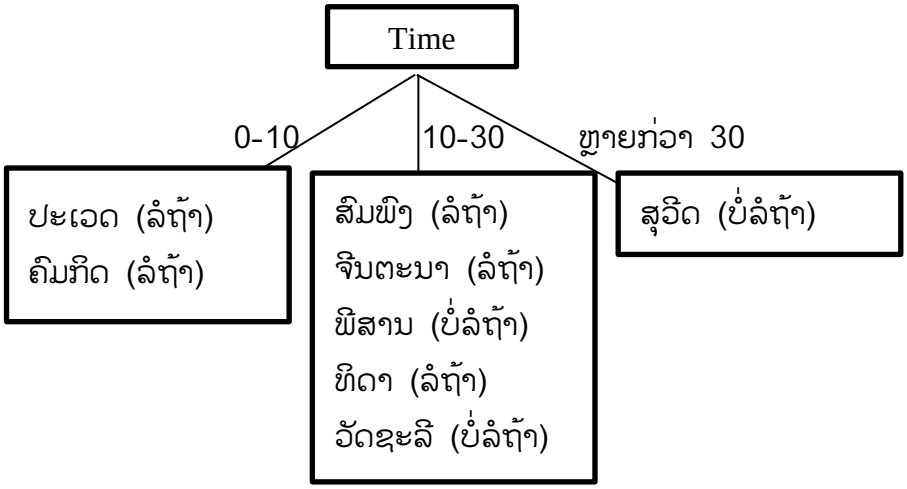
$$\mathrm{Gain(Type)} = (0.4237 + 0.5306) - \left(\frac{1}{2}(0.3112 + 0.5) + \frac{1}{4}(0.5 + 0.5)\frac{1}{4}(0.5 + 0.5)\right)$$

$$\mathrm{Gain(Type)} = (0.9543) - \left(\frac{1}{2}(0.8112) + \frac{1}{4}(1)\frac{1}{4}(1)\right)$$

$$\mathrm{Gain(Type)} = (0.9543) - (0.4056 + 0.25 + 0.25)$$

$$\mathrm{Gain(Type)} = 0.9543 - 0.9056 = 0.0487$$

ໄລຍະເວລາໃນການລໍຖ້າ (Time).



ຮູບທີ 5.6 ແບບຈຳລອງໄລຍະເວລາໃນການລໍຖ້າ

ຈາກແບບຈຳລອງໄລຍະເວລາໃນການລໍຖ້າ ເມື່ອແທນໃນ Gain Function ຈະໄດ້ດັ່ງນີ້:

$$\mathrm{E(s)} = -\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}$$

ດັ່ງນັ້ນຈະໄດ້

$$\mathrm{Gain(Time)} = \left(-\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}\right) - \left(\frac{2}{8}\left(-\frac{2}{2}log_2\frac{2}{2}\right) + \frac{5}{8}\left(-\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5}\right) + \frac{1}{8}\left(-\frac{1}{1}log_2\frac{1}{1}\right)\right)$$

$$\mathrm{Gain(Time)} = \left(-\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}\right) - \left(\frac{1}{4}(-log_2 1) + \frac{5}{8}\left(-\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5}\right) + \frac{1}{8}(-log_2 1)\right)$$

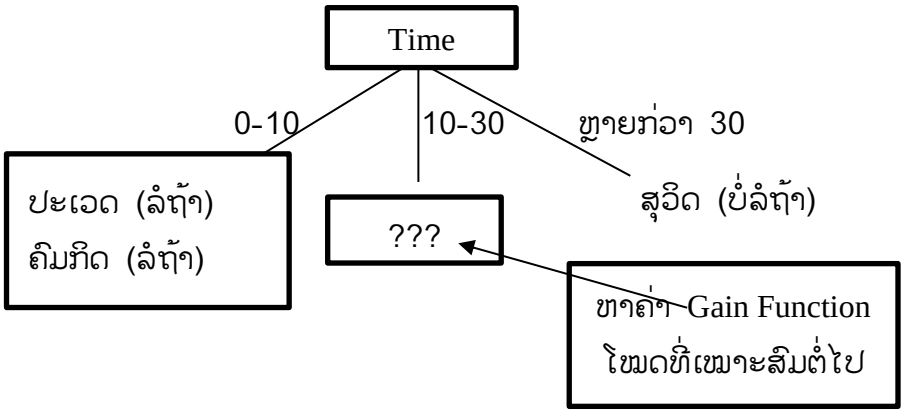
$$\mathrm{Gain(Time)} = (0.4237 + 0.5306) - \left(\frac{1}{4}(0) + \frac{5}{8}(0.4421 + 0.5287) + \frac{1}{8}(0)\right)$$

$$\mathrm{Gain(Time)} = (0.9543) - \left(\frac{5}{8}(0.9708)\right)$$

$$\mathrm{Gain(Time)} = 0.9543 - 0.6067 = 0.39708$$

ຈາກການຄຳນວນຄ່າ Gain Function ຂອງແຕ່ລະຊຸດຂໍ້ມູນຈະໄດ້ຄ່າທີ່ແຕກຕ່າງກັນໄປ, ໂດຍຄ່າທີ່ຫຼາຍທີ່ສຸດຈະສາມາດບົ່ງບອກໄດ້ວ່າຊຸດຂໍ້ມູນນັ້ນມີຄວາມເໝາະສົມໃນການນຳມາເປັນໂໜດເລີ່ມຕົ້ນ ໃນການສ້າງ Decision Tree ເຊິ່ງກໍ່ແມ່ນ Gain (Time) ທີ່ມີຄ່າຫຼາຍທີ່ສຸດ, ຈາກນັ້ນຈຶ່ງພິຈາລະນາໂໜດທີ່ເໝາະສົມໃນອັນດັບຕໍ່ໄປໂດຍການຄຳນວນຫາ Gain Function ເຊັ່ນດຽວກັນກັບຂັ້ນຕອນທີ່ຜ່ານມາ, ແຕ່ຈະພິຈາລະນາພຽງໂໜດທີ່ເຫຼືອ ເຊິ່ງຈະເຊື່ອມຕໍ່ກັບໂໜດທີ່ເລີ່ມຕົ້ນ ໂດຍພິຈາລະນາຈາກກິ່ງກ້ານທີ່ຍັງມີ

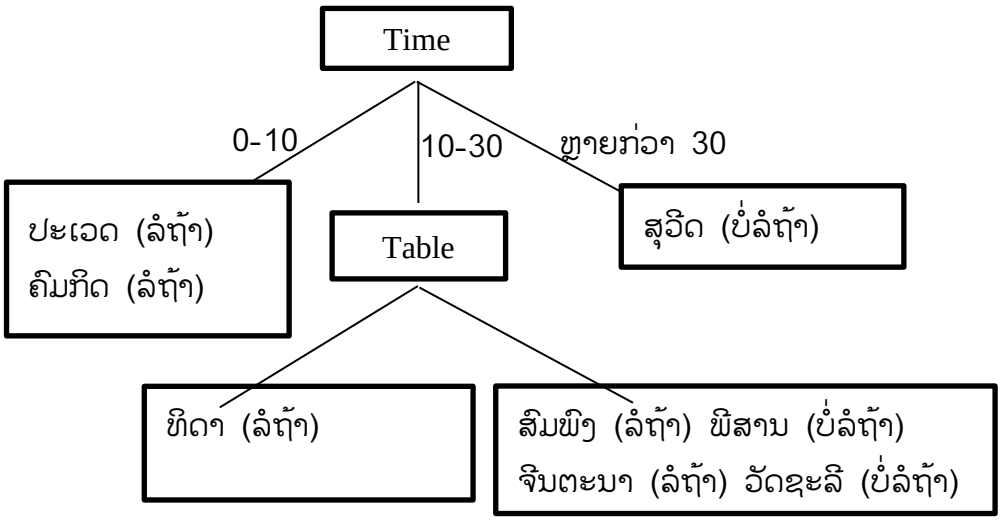
ຄວາມຊັບຊ້ອນຂອງຂໍ້ມູນ ຫຼື ຍັງຈຳແນກໄດ້ບໍ່ໝົດ, ໂໜດນັ້ນຍັງຕ້ອງແຕກກິ່ງກ້ານຕໍ່ໄປເພື່ອແຍກຂໍ້ມູນທີ່ແຕກຕ່າງກັນອອກເປັນກຸ່ມທີ່ເໝາະສົມ. (ດັ່ງຮູບທີ 5.7 ໂໜດເຄິ່ງກາງຍັງມີຄວາມຕ້ອງການຂອງລູກຄ້າ ລໍຖ້າ ແລະ ບໍ່ລໍຖ້າ ປົນກັນຢູ່)



ຮູບທີ 5.7 ສະແດງການຫາຄ່າໂໜດລຳດັບຕໍ່ໄປໃນ Decision Tree

ດັ່ງນັ້ນຈຶ່ງຕ້ອງພິຈາລະນາຄ່າ Gain Function ຂອງຊຸດຂໍ້ມູນທີ່ເຫຼືອ ດັ່ງນີ້:

ສະຖານະຂອງໂຕະ (Table)



ຮູບທີ 5.8 ແບບຈຳລອງປະລິມານໂຕະວ່າງ

ຈາກແບບຈຳລອງປະລິມານໂຕະວ່າງ ເມື່ອແທນໃນ Gain Function ຈະໄດ້ດັ່ງນີ້:

$$\mathrm{E(s)} = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5}$$

ດັ່ງນັ້ນຈະໄດ້

$$\mathrm{Gain(Table)} = \left(-\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5}\right) - \left(\frac{1}{5}\left(-\frac{1}{1}log_2\frac{1}{1}\right) + \frac{4}{5}\left(-\frac{2}{4}log_2\frac{2}{4} - \frac{2}{4}log_2\frac{2}{4}\right)\right)$$

$$\text{Gain(Table)} = \left(-\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5}\right) - \left(\frac{1}{5}(-log_2 1) + \frac{4}{5}\left(-\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2}\right)\right)$$

$$\text{Gain(Table)} = (0.4421 + 0.5257) - \left(\frac{1}{5}(0) + \frac{4}{5}(0.5 + 0.5)\right)$$

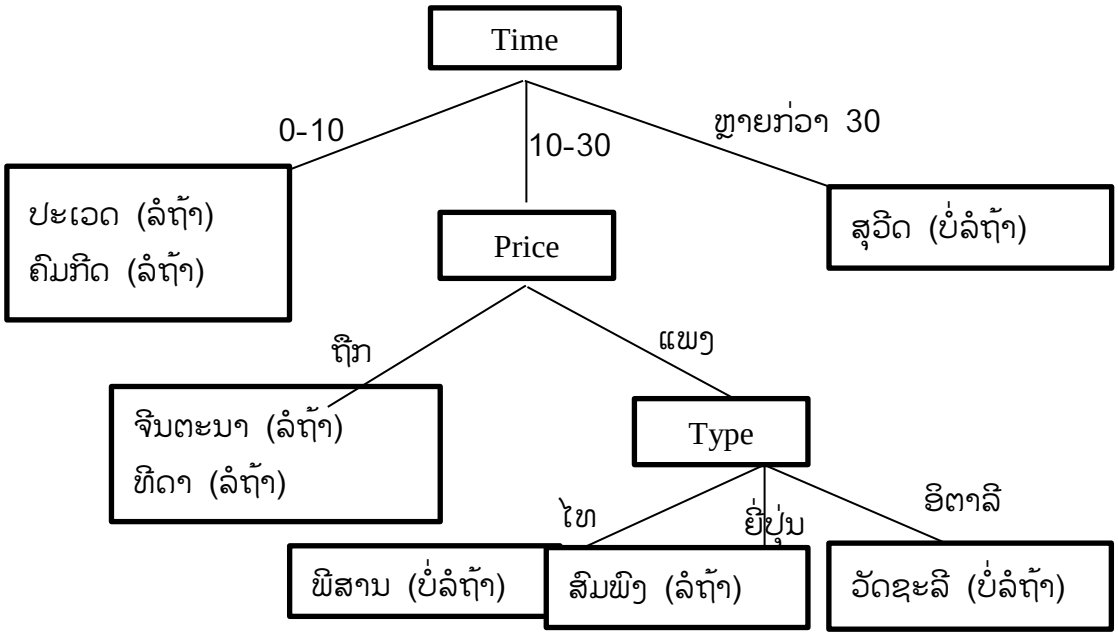
$$\text{Gain(Table)} = (0.9708) - \left(\frac{4}{5}(1)\right)$$

$$\text{Gain(Time)} = 0.9543 - 0.80 = 0.1708$$

ນອກຈາກຊຸດຂໍ້ມູນປະລິມານໂຕະອ່າງແລ້ວຈຳເປັນຕ້ອງຄຳນວນຄ່າ Gain Function ຂອງຊຸດຂໍ້ມູນອື່ນດ້ວຍ (Price, Type) ເຊິ່ງໄດ້ຄ່າດັ່ງນີ້:

Gain (Table) = 0.1708 Gain (Price) = 0.4199 Gain (Type) = 0.4199

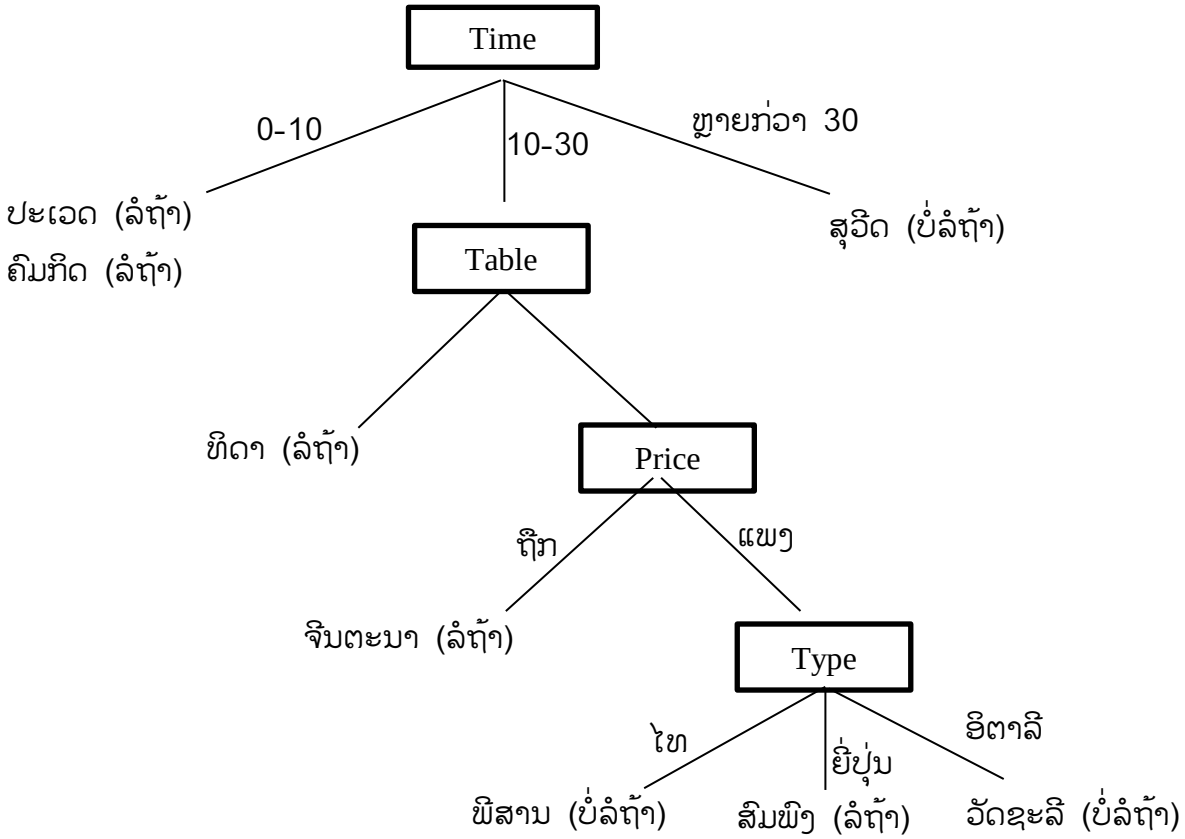
ໃນກໍລະນີທີ່ຄ່າຂອງ Gain Function ທີ່ເໝາະສົມເທົ່າກັນຈະສາມາດເລືອກຊຸດຂໍ້ມູນຊຸດໃດກໍ່ໄດ້ ເນື່ອງຈາກຊຸດຂໍ້ມູນທັງສອງສາມາດຄຳນວນຂໍ້ມູນໄດ້ຢ່າງສົມບູນຄືກັນ ພຽງແຕ່ຈະມີໂຄງສ້າງຂອງ Decision Tree ທີ່ແຕກຕ່າງກັນເທົ່ານັ້ນສຳລັບຕົວຢ່າງນີ້ຈະເລືອກ Price ເປັນໂໜດຕໍ່ໄປ ເຊິ່ງຈະໄດ້ Decision Tree ດັ່ງຮູບທີ 5.9



ຮູບທີ 5.9 ສະແດງ Decision Tree ທີ່ສົມບູນຂອງຕົວຢ່າງທີ 5.3

ຮູບທີ 5.9 ຈະເປັນ Decision Tree ທີ່ມີຄວາມສົມບູນແລ້ວ ເນື່ອງຈາກສາມາດໄຈແຍກຂໍ້ມູນທີ່ສົນໃຈອອກຈາກຊຸດຂໍ້ມູນທັງໝົດໄດ້ຢ່າງຈະແຈ້ງ, ເຖິງວ່າຈະມີໂໜດບໍ່ຄົບຕາມຈຳນວນຂອງຊຸດຂໍ້ມູນທັງໝົດທີ່ພິຈາລະນາກ່ຕາມ. ເນື່ອງຈາກ Decision Tree ທີ່ໄດ້ບໍ່ຈຳເປັນຕ້ອງເອົາໂໜດ Table ມາພິຈາລະນາ. ດ້ວຍເຫດນີ້ຈຶ່ງມີຄວາມຊັບຊ້ອນນ້ອຍກ່ວາ Decision Tree ທີ່ເອົາໂໜດ Table ມາຮ່ວມ

ພິຈາລະນາ. ໃນຄວາມເປັນຈິງອາດມີ Decision Tree ທີ່ສອດຄ່ອງກັບຊຸດຂໍ້ມູນທັງໝົດໄດ້ຫຼາຍກ່ວາໜຶ່ງ ຮູບແບບ, ເຊິ່ງອາດເປັນ Decision Tree ທີ່ງ່າຍຕໍ່ການພິຈາລະນາຂໍ້ມູນ ຫຼື ອາດຊັບຊ້ອນກ່ວາ Decision Tree ເກົ່າກໍໄດ້ ດັ່ງຮູບທີ 5.10

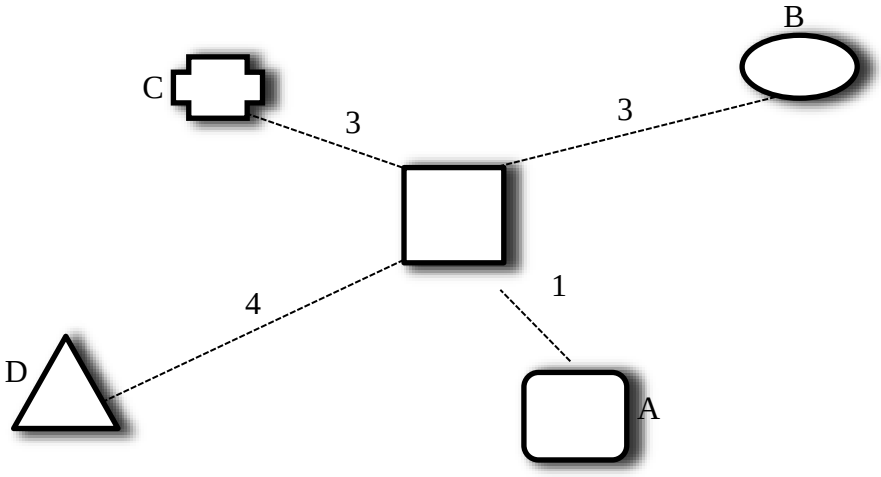


ຮູບທີ 5.10 ສະແດງຕົວຢ່າງ Decision Tree ອີກໜຶ່ງຮູບແບບທີ່ສອດຄ່ອງກັບ ຊຸດຂໍ້ມູນຂອງຕົວຢ່າງທີ 5.3

ຮູບທີ 5.9 ແລະ 5.10 ຈະເຫັນໄດ້ວ່າ Decision Tree ທັງໝົດແຕກຕ່າງກັນ, ໃນຮູບທີ 5.10 ຈະມີຈຳນວນໂໜດຂອງໃບຫຼາຍກ່ວາ ແລະ ສາມາດຈຳແນກຂໍ້ມູນອອກຈາກຊຸດຂໍ້ມູນໄດ້ຄືກັນກັບຮູບທີ 5.9 ແຕ່ຄວາມຊັບຊ້ອນ ຫຼື ຈຳນວນໃບຫຼາຍກ່ວາ ອາດເຮັດໃຫ້ການຕັດສິນໃຈຊ້າລົງໄດ້. ຖ້າພິຈາລະນາຮູບທີ 5.9 ແລ້ວຈະເຫັນໄດ້ວ່າປັດໄຈສຳຄັນທີ່ສຸດຂອງພຶດຕິກຳການລໍຖ້າຂອງລູກຄ້າທັງ 8 ຄົນ ຈະເປັນເລື່ອງ ຂອງເວລາ, ດັ່ງນັ້ນສະຖານະຂອງໂຕະຈຶ່ງບໍ່ຈຳເປັນຖ້າລູກຄ້າຮູ້ໄລຍະເວລາທີ່ແນ່ນອນວ່າຕ້ອງລໍຖ້າດົນ ປານໃດ. ເມື່ອພິຈາລະນາທີ່ໂໜດທຳອິດລົງມາຄືລາຄາ ຖືວ່າເປັນປັດໄຈທີ່ຊ່ວຍຈຳແນກຂໍ້ມູນໄດ້ມີ ປະສິດທິພາບຫຼາຍກ່ວາຊຸດຂໍ້ມູນສະຖານະຂອງໂຕະ, ເນື່ອງຈາກວ່າສາມາດຈຳແນກລູກຄ້າທີ່ມີຄວາມ ຕ້ອງການລໍຖ້າໄດ້ 2 ຄົນ ໃນຄະນະທີ່ໂໜດສະຖານະຂອງໂຕະສາມາດຈຳແນກໄດ້ພຽງ 1 ຄົນເທົ່ານັ້ນ. ຖ້າເປັນລະບົບທີ່ມີຂໍ້ມູນຈຳນວນຫຼາຍອາດຈະເຮັດໃຫ້ເສຍເວລາໄປໂດຍບໍ່ມີປະໂຫຍດ. ດັ່ງນັ້ນໃນການສ້າງ Decision Tree ຈຳເປັນຕ້ອງຄຳນວນຫາຊຸດຂໍ້ມູນທີ່ເໝາະສົມທີ່ສຸດ ເພື່ອໃຫ້ໄດ້ Decision Tree ທີ່ສົມບູນ ແລະ ມີປະສິນຕິພາບເມື່ອເອົາໄປໃຊ້ໃນລະບົບການຮຽນຮູ້ຂອງປັນຍາປະດິດ.

5.6. Nearest Neighbor Classification

ເປັນການຮຽນຮູ້ປະເພດ Unsupervised Learning ເຊິ່ງເປັນການຈຳແນກ ຫຼື ຈັດກຸ່ມທີ່ມີວິທີການບໍ່ຊັບຊ້ອນ, ໂດຍພິຈາລະນາຊຸດຂໍ້ມູນທີ່ໃກ້ຄຽງກ່ວາຄື: ຂໍ້ມູນມີຄ່າທີ່ໃກ້ຄຽງກັບຄ່າຂອງຂໍ້ມູນທີ່ພິຈາລະນາຫຼາຍທີ່ສຸດ, ໃນນີ້ຄ່າຄວາມໃກ້ຄຽງຈະໝາຍເຖິງໄລຍະທາງ (Distance) ທີ່ມີຄ່ານ້ອຍທີ່ສຸດລະຫວ່າງຊຸດຂໍ້ມູນກັບຂໍ້ມູນທີ່ພິຈາລະນາ ຂໍ້ມູນດັ່ງກ່າວນີ້ເອີ້ນວ່າ: “Nearest Neighbor” ດັ່ງຮູບທີ 5.11



ຮູບທີ 5.11 ສະແດງຕົວຢ່າງໄລຍະເວລາລະຫວ່າງຊຸດຂໍ້ມູນ [*Jones, 2008*]

ຮູບທີ 5.11 ຊຸດຂໍ້ມູນທີ່ມີໄລຍະທາງໃກ້ກັບສູນກາງຫຼາຍທີ່ສຸດຄື A ເຊິ່ງມີໄລຍະເປັນ 1 ໃນທິດສະດີຂອງ Nearest Neighbor ການຈຳແນກຊຸດຂໍ້ມູນທີ່ມີໄລຍະຫ່າງ 1 ຈະເອີ້ນວ່າ: “1NN (One Nearest Neighbor)” ໂດຍໄລຍະຫ່າງຂອງຂໍ້ມູນນັ້ນສາມາດກຳນົດໄດ້ວ່າຕ້ອງການໜ້ອຍຫຼາຍເທົ່າໃດ, ດ້ວຍເຫດນີ້ຈຶ່ງມີການຈຳແນກທີ່ເອີ້ນວ່າ: “k-NN” ເຊິ່ງ k ແທນດ້ວຍຄ່າໄລຍະທາງລະຫວ່າງຂໍ້ມູນທີ່ຕ້ອງການ ສຳລັບການຫາຄ່າໄລຍະທາງຈະໃຊ້ສົມຜົນຈາກທິດສະດີການວັດຄ່າຂອງ Euclidean ດັ່ງນີ້.

$$\mathrm{d} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

ສົມຜົນດັ່ງກ່າວເປັນການຫາໄລຍະທາງລະຫວ່າງຊຸດຂໍ້ມູນ ໂດຍແທນຄ່າທີ່ບັນຈຸຢູ່ພາຍໃນຊຸດຂໍ້ມູນທັງສອງ, ເຊິ່ງ p ແມ່ນຄ່າໃນຊຸດຂໍ້ມູນທີ່ຕ້ອງການຈຳແນກ ມີຄ່າຕັ້ງແຕ່ $(p_1, p_2, p_3, \dots, p_n)$ ແລະ q ແມ່ນຄ່າໃນຊຸດຂໍ້ມູນຂ້າງຄຽງທີ່ນຳມາພິຈາລະນາ ມີຄ່າຕັ້ງແຕ່ $(\mathrm{q}_1, \mathrm{q}_2, \mathrm{q}_3, \dots, \mathrm{q}_n)$ ຂຶ້ນຢູ່ກັບຈຳນວນຂໍ້ມູນທີ່ຢູ່ພາຍໃນຊຸດຂໍ້ມູນນັ້ນ ດັ່ງຕົວຢ່າງທີ 5.4

ຕົວຢ່າງທີ 5.4

ກຳນົດໃຫ້ມີການຈຳແນກສັດ 5 ຊະນິດ ຈຳແນກຄຸນສົມບັດ ທີ່ສາມາດບົ່ງບອກເຖິງສາຍພັນໄດ້ ດັ່ງນີ້:

	ออกลูกเป็นไข่	ออกลูกเป็นไข่ ล้ຽงลูกด้วยน้ำนม	มีขนปกคลุมร่างกาย ส่วนมีหัด ปีก ส่วนด้าน	สัดเลือดเย็น	สัดเลือดเย็น สัดมีปีก และ น้ำ หายใจด้วยเหงือก		
ภึบ		1				1	1			สัดเถิงบึกเถิงน้ำ
เปัด		1		1		1		1		สัดจำพอกนึก
เจย	1		1	1						สัดล้ำงลูกด้อยน้ำนึม
ๆ		1			1		1			สัดเลือยคาน
ปาซามอน		1			1		1		1	สัดจำพอกปา

ມີชุดข้ำมูนขอๆสัดชะนิดหึ่งรู้พุงๆข้ำมูนด้านคุนสึมบัด ดั่งนี้

	ออกลูกเป็นไข่	ออกลูกเป็นไข่ ล้ຽงลูกด้วยน้ำนม	มีขนปกคลุมร่างกาย ส่วนมีหัด ปีกส่วนด้าน	สัดเลือดเย็น	สัดเลือดเย็น สัดมีปีก และ น้ำ หายใจด้วยเหงือก	สายพันธุ์
สัดปิดสะหฯา		1					1		???

ໂดยการรู้ว่าสัดชะนิดนี้เป็นสัดที่ใก้ๆกับสายพันใดหลายที่สุด (1NN) จากชุดข้ำมูนขอๆ สัดปิดสะหฯา จะเอิามาใช้หาค่าไลยะทางขอๆชุดข้ำมูนสัดทัง 5 ชะนิด, ด้อยสึมผึน Euclidean เมื่อแทนค่าในแต่ละชุดข้ำมูนได้ดั่งนี้:

ไลยะทางขอๆชุดข้ำมูนภึบ (d_1)

$$d_1 = \sqrt{(0)^2 + (1-1)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0-1)^2 + (1-1)^2 + (0)^2} = \sqrt{1} = 1$$

ไลยะทางขอๆชุดข้ำมูนเปัด (d_2)

$$d_2 = \sqrt{(0)^2 + (1-1)^2 + (0)^2 + (0-1)^2 + (0)^2 + (0-1)^2 + (0)^2 + (1-1)^2 + (0)^2}$$
$$= \sqrt{1+1} = \sqrt{2} = 1.414$$

ไลยะทางขอๆชุดข้ำมูนเจย (d_3)

$$d_3 = \sqrt{(0-1)^2 + (1-0)^2 + (0-1)^2 + (0-1)^2 + (0)^2 + (0)^2 + (0)^2 + (1-0)^2 + (0)^2}$$
$$= \sqrt{1+1+1+1+1} = \sqrt{5} = 2.236$$

ໄລຍະທາງຂອງຊຸດຂໍ້ມູນງູ (d_4)

$$d_4 = \sqrt{(0)^2 + (1-1)^2 + (0)^2 + (0)^2 + (0-1)^2 + (0)^2 + (0-1)^2 + (1-0)^2 + (0)^2}$$
$$= \sqrt{1+1+1} = \sqrt{3} = 1.732$$

ໄລຍະທາງຂອງຊຸດຂໍ້ມູນປາຊາມອນ (d_5)

$$d_5 = \sqrt{(0)^2 + (1-1)^2 + (0)^2 + (0)^2 + (0-1)^2 + (0)^2 + (0-1)^2 + (1-0)^2 + (0-1)^2}$$
$$= \sqrt{1+1+1+1} = \sqrt{4} = 2$$

ຈາກຄ່າຂອງໄລຍະທາງທີ່ໄດ້ຈາກຊຸດຂໍ້ມູນສັດທັງ 5 ຊະນິດ ໄລຍະທາງທີ່ມີຄ່າໜ້ອຍທີ່ສຸດຄື d_1 ສະແດງວ່າສັດປີດສະໜາມີຂໍ້ມູນທີ່ໃກ້ຄຽງກັບຊຸດຂໍ້ມູນຂອງກົບ. ຈຶ່ງສະຫຼຸບໄດ້ວ່າ: ສັດປີດສະໜາຊະນິດນີ້ ມີສາຍພັນເປັນ ***ສັດເຄິ່ງບົກເຄິ່ງນ້ຳ.***

ຈາກຕົວຢ່າງທີ 5.4 ເປັນການຄຳນວນຫາໄລຍະທີ່ໃກ້ທີ່ສຸດຄື ຈະມີຊຸດຂໍ້ມູນຂ້າງຄຽງທີ່ມີໄລຍະຫ່າງ ນ້ອຍທີ່ສຸດພຽງໜຶ່ງຊຸດຂໍ້ມູນເທົ່ານັ້ນ, ໃນກໍລະນີທີ່ຕ້ອງການຊຸດຂໍ້ມູນທີ່ມີຄ່າໃກ້ຄຽງຫຼາຍກ່ວາໜຶ່ງຊຸດຂໍ້ມູນ ຫຼື k ຊຸດ, ການຈຳແນກຫາຊຸດຂໍ້ມູນປະເພດນີ້ຈະເອີ້ນວ່າ: "k-NN" ດັ່ງຕໍ່ໄປນີ້.

ຕົວຢ່າງທີ 5.5

ກຳນົດໃຫ້ມີການຈຳແນກສັດ 5 ຊະນິດ ຈຳແນກຄຸນສົມບັດ ທີ່ສາມາດບົ່ງບອກເຖິງສາຍພັນໄດ້ດັ່ງນີ້:

	ออกลูกเป็นไข่	ออกลูกเป็นไข่	ล้ำลูกด้วยน้ำนม	มีเขี้ยวงอกยาว	มีหนังหุ้มตัว หรือมีเกล็ด	สัดเลือดเย็น	สัดเลือดเย็น ตัวอยู่ทั้งน้ำ และ บก	หายใจด้วยเหงือก	สายพันธุ์
ກົບ		1					1	1		สัตเคิ่งบกเคิ่งน้ำ
เป็ด		1		1		1		1		สัตจำพวกนก
เจย	1		1	1						สัตล้ำลูกด้วยน้ำนม
งู		1			1		1			สัตเลื้อยคาน
ปาแซมอน		1			1		1		1	สัตจำพวกปา
แข้		1			1		1			สัตเลื้อยคาน
แมว	1		1	1		1				สัตล้ำลูกด้วยน้ำนม
นกกะจอกเทด	1			1		1				สัตจำพวกนก

ມີຊຸດຂໍ້ມູນຂອງສັດຊະນິດໜຶ່ງຮູ້ພຽງຂໍ້ມູນດ້ານຄຸນສົມບັດ ດັ່ງນີ້:

	ອອກລູກເປັນໂຕ	ອອກລູກເປັນໄຂ່	ລ້ຽງລູກດ້ວຍນ້ຳນົມ	ມີຂົນຕາມຮ່າງກາຍ	ຕົວມີເກັດ ຫຼືຜິວດ້ານ	ສັດເລືອດອຸ່ນ	ສັດເລືອດເຢັນ	ສັດຢູ່ບົກ ແລະ ນ້ຳ	ຫາຍໃຈດ້ວຍເຫືອກ	ສາຍພັນ
ສັດປີດສະໜາ			1	1		1				???

ຕ້ອງການຮູ້ວ່າສັດຊະນິດນີ້ເປັນສັດທີ່ໃກ້ຄຽງກັບສາຍພັນໃດແດ່ຢ່າງນ້ອຍ 3 ຊະນິດ (3NN) ຫາຄ່າໄລຍະທາງຂອງແຕ່ລະຊຸດຂໍ້ມູນ (ຄືຕົວຢ່າງທີ 5.4) ຈະໄດ້ຜົນຮັບດັ່ງນີ້:

ກົບ (d_1) = 2.449

ເປັດ (d_2) = 1.732

ເຈຍ (d_3) = 1.414

ງູ (d_4) = 2.236

ປາຊາມອນ (d_5) = 2.645

ແຂ້ (d_6) = 2.236

ແມວ (d_7) = 1

ນົກກະຈອກເທດ (d_8) = 1.414

ຈາກຄ່າໄລຍະທາງທີ່ໄດ້ຂອງທັງ 8 ຊຸດຂໍ້ມູນ ຈະເລືອກຄ່າໄລຍະທາງທີ່ນ້ອຍທີ່ສຸດ 3 ອັນດັບ ໄດ້ແກ່ ແມວ (d_7) = 1, ເຈຍ (d_3) = 1.414 ແລະ ນົກກະຈອກເທດ (d_8) = 1.414 ດັ່ງນັ້ນສັດປີດສະໜາດັ່ງກ່າວມີຄວາມໃກ້ຄຽງກັບສາຍພັນສັດລ້ຽງລູກດ້ວຍນ້ຳນົມ ແລະ ສັດຈຳພວກນົກ. ແຕ່ວ່າທັງ 3 ຊຸດຂໍ້ມູນທີ່ໄດ້ມາມີ 2 ຊຸດຂໍ້ມູນທີ່ມີສາຍພັນຄືກັນ, ຈຶ່ງສາມາດສະຫຼຸບຈາກຂໍ້ມູນທີ່ໄດ້ວ່າສັດປີດສະໜາຊະນິດນີ້ມີສາຍພັນເປັນ **ສັດລ້ຽງລູກດ້ວຍນ້ຳນົມ.**

ຈະເຫັນວ່າ Decision Tree ແລະ Nearest Neighbor Classification ເປັນເຕັກນິກການຮຽນຮູ້ຂອງເຄື່ອງຈັກທີ່ແຕກຕ່າງກັນ, ໂດຍ Decision Tree ຈະເປັນວິທີການຈຳແນກເພື່ອຫາຄວາມຖືກຕ້ອງຂອງຂໍ້ມູນທີ່ສົນໃຈ, ໂດຍຮູ້ຄ່າຂອງຊຸດຂໍ້ມູນທັງໝົດ ເຊິ່ງເປັນໄປຕາມຫຼັກການຂອງ Supervised Learning ສ່ວນ, Nearest Neighbor ຈະພິຈາລະນາຈາກການທຳນາຍຄ່າຂອງຊຸດຂໍ້ມູນ ເຊິ່ງເປັນວິທີການແບບ Unsupervised Learning ໂດຍໃຫ້ຄວາມສຳຄັນກັບຄວາມສຳພັນຂອງຊຸດຂໍ້ມູນແທນການຄຳນວນຄ່າຂອງຄວາມນ່າຈະເປັນຈາກຂໍ້ມູນທັງໝົດ. ດັ່ງນັ້ນ, ຈຶ່ງຕ້ອງພິຈາລະນາຊຸດຂໍ້ມູນທີ່ສົນໃຈວ່າຄວນເລືອກໃຊ້ວິທີການຈຳແນກກຸ່ມແບບໃດຈຶ່ງຈະເໝາະສົມທີ່ສຸດ.

ສະຫຼຸບ:

ການຮຽນຮູ້ຂອງເຄື່ອງຈັກ (Machine Learning) ໝາຍເຖິງເຕັກນິກ ຫຼື ຂະບວນການທີ່ໃຊ້ສຳລັບ ປັບແຕ່ງ, ປຸງແຕ່ງເຄື່ອງຈັກ ຫຼື ຄອມພິວເຕີໃຫ້ມີພຶດຕິກຳສະເພາະຕ໊ອທີ່ສະໜັບສະໜຸນການຮຽນຮູ້ ແລະ ສຶກສາຂໍ້ມູນຈາກປະສົບການ. ເປັນວິທີທີ່ຊ່ວຍໃຫ້ເຄື່ອງຈັກສາມາດຮຽນຮູ້ໄດ້ຄືກັບມະນຸດ. ໂດຍຈະ ກ່ຽວຂ້ອງກັບການພັດທະນາ ແລະ ອອກແບບອານກໍລິດທຶມ (Algorithm) ຫຼື ເຕັກນິກຕ່າງໆທີ່ຊ່ວຍໃຫ້ ເຄື່ອງຈັກ ຫຼື ຄອມພິວເຕີມີຄວາມສາມາດໃນການຮຽນຮູ້ຂໍ້ມູນທີ່ສົນໃຈ ແລະ ເອົາໄປໃຊ້ໃນການແກ້ໄຂ ບັນຫາສະເພາະໜ້າໄດ້, ພ້ອມທັງເກັບສະສົມໄວ້ເປັນຄວາມຮູ້ ເຊິ່ງສາມາດເອົາໄປໃຊ້ໃນການແກ້ໄຂບັນຫາ ທີ່ອາດເກີດຂຶ້ນໄດ້ໃນອານາຄົດ. ການຮຽນຮູ້ສາມາດແບ່ງໄດ້ເປັນ 2 ຊະນິດຄື: 1) Deductive ເປັນການ ຮຽນຮູ້ໂດຍອາໄສຄວາມຮູ້ທີ່ເປັນຈິງຢູ່ແລ້ວ ມີຄວາມຈິງເປັນສາກົນເຊິ່ງທຸກຄົນຍອມຮັບ, ໂດຍສາມາດຄາດ ການໄດ້ວ່າເຫດການດັ່ງກ່າວຈະເກີດຂຶ້ນແນ່ນອນຕາມຮູບແບບຂອງສິ່ງແວດລ້ອມ ແລະ 2) Inductive ເປັນການຮຽນຮູ້ຈາກເຫດການ ຫຼື ສິ່ງທີ່ສົນໃຈ, ໂດຍຮູ້ຂໍ້ມູນພຽງບາງສ່ວນແລ້ວຈຶ່ງເອົາໄປສຶກສາ ແລະ ສ້າງຄວາມເຂົ້າໃຈ ເພື່ອໄດ້ຂໍ້ມູນທັງໝົດຈົນກາຍເປັນຄວາມຈິງສາກົນທີ່ຄົນສ່ວນຫຼາຍຍອມຮັບ.

Ensemble Learning ເປັນທິດສະດີທີ່ຊ່ວຍໃຫ້ປັນຍາປະດິດທີ່ພັດທະນາຂຶ້ນສາມາດລວບລວມ ແລະ ເລືອກທີ່ຈະວິເຄາະສົມມຸດຖານ ເພື່ອຄັດເລືອກຂໍ້ມູນທີ່ເໝາະສົມ ແລະ ມີປະສິດທິພາບ, ເຊິ່ງໃຊ້ ການຄາດເດົາ ຫຼື ພະຍາກອນໂດຍວິເຄາະຈາກສົມມຸດຖານທີ່ກ່ຽວຂ້ອງທັງໝົດ ເພື່ອໃຊ້ເປັນແນວທາງໃນ ການແກ້ໄຂບັນຫາ ຫຼື ຮຽນຮູ້ສິ່ງທີ່ລະບົບສົນໃຈ. ເນື່ອງຈາກຂໍ້ມູນໜຶ່ງຊຸດອາດສ້າງເປັນ Decision Tree ຈຳນວນຫຼາຍ ຈຶ່ງຕ້ອງຈັດກຸ່ມຄັດເລືອກຕ໊ອຢ່າງທີ່ດີທີ່ສຸດ ແລະ ທີ່ມີຄວາມເປັນໄປໄດ້ໃນການນຳໄປ ວິເຄາະ ຊ່ວຍໃຫ້ການຮຽນຮູ້ມີປະສິດທິພາບສູງຂຶ້ນລວມເຖິງສາມາດແກ້ໄຂບັນຫາທີ່ມີຄວາມຊັບຊ້ອນຫຼາຍ ໄດ້.

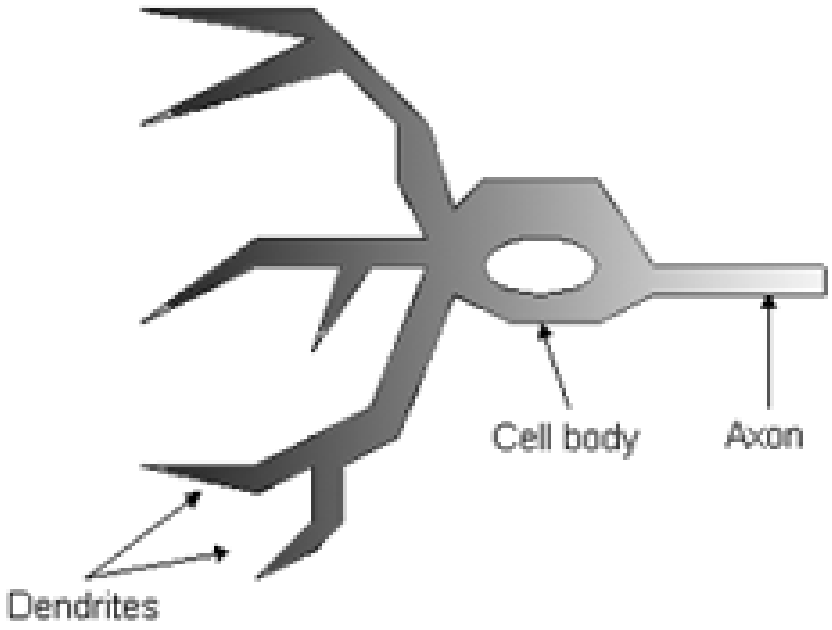
Decision Tree ເປັນເຕັກນິກໜຶ່ງຂອງການຮຽນຮູ້ຂອງເຄື່ອງຈັກ ທີ່ໃຊ້ໃນການພັດທະນາການຮຽນຮູ້ ຂອງເຄື່ອງຈັກ ຫຼື ເຄື່ອງຄອມພິວເຕີ, ເປັນແບບຈຳລອງທີ່ໃຊ້ສຳລັບການຄາດຄະເນ ຫຼື ຄາດເດົາ ເຫດການທີ່ຈະເກີດຂຶ້ນ ບາງຄັ້ງອາດເອີ້ນວ່າ: “Classification Tree” ປະກອບດ້ວຍໃບ (Leaf) ເປັນ ສ່ວນຂອງຂໍ້ມູນທີ່ເຮົາສົນໃຈ, ແຕ່ລະໃບຖືກເຊື່ອມດ້ວຍກິ່ງກ້ານ (Branch) ເຊິ່ງເປັນຂໍ້ມູນທີ່ໃຊ້ເຊື່ອມຕໍ່ໃບ ໃສ່ກັນ, ກຳເນີດຈາກສ່ວນເທິງສຸດຄື ຮາກ (Root). ສຳລັບການສ້າງແບບຈຳລອງ Decision Tree ຈະ ກ່ຽວຂ້ອງກັບກຸ່ມຂອງຂໍ້ມູນ ໂດຍມີຂະບວນການສ້າງທີ່ສາມາດຄຳນວນແບບກັບຄືນ ຫຼື ເຮັດຊ້ຳຂັ້ນຕອນ ເກົ່າໄດ້, ການສ້າງຈະພິຈາລະນາໃຫ້ໂໜດເລີ່ມຕົ້ນ ຫຼື ຮາກ “Root” ເປັນອັນດັບທຳອິດ ແລະ ກິ່ງກ້ານທີ່ ແຕກແໜ່ງຕໍ່ໄປ, ເຊິ່ງອາໄສການຄຳນວນຫາຄ່າ “Information Gain” ທີ່ປະກອບດ້ວຍຄ່າຂອງຂໍ້ມູນອີກ ຊະນິດໜຶ່ງທີ່ເອີ້ນວ່າ: “Entropy” ເພື່ອໃຊ້ປຽບທຽບ ແລະ ຄັດເລືອກຊຸດຂໍ້ມູນທີ່ເໝາະສົມທີ່ສຸດ.

Nearest Neighbor Classification ເປັນວິທີການຈຳແນກຂໍ້ມູນ ໂດຍພິຈາລະນາເຖິງຄວາມສຳພັນ ຂອງຊຸດຂໍ້ມູນ ແລະ ຄ່າໃນຊຸດຂໍ້ມູນທີ່ຕ້ອງການຈຳແນກ, ວິທີການນີ້ຈະເຮັດການຄຳນວນໄລຍະທາງ ລະຫວ່າງຂໍ້ມູນຂ້າງຄຽງ ເພື່ອໃຫ້ຮູ້ວ່າຊຸດຂໍ້ມູນໃດມີຄວາມໃກ້ຄຽງກັນຫຼາຍທີ່ສຸດ. ຄວາມໃກ້ຄຽງດັ່ງກ່າວ ສາມາດນຳມາສະຫຼຸບໄດ້ວ່າຂໍ້ມູນທີ່ເຮັດການຈຳແນກຈັດຢູ່ໃນກຸ່ມໃດ, ການຄຳນວນຊຸດຂໍ້ມູນໃກ້ຄຽງອາດ ຄັດເລືອກສະເພາະຊຸດຂໍ້ມູນທີ່ມີຄ່າໃກ້ຄຽງຫຼາຍທີ່ສຸດພຽງໜຶ່ງຄ່າ ຫຼື ຊຸດຂໍ້ມູນທີ່ໃກ້ຄຽງຫຼາຍໆຊຸດມາ ປະກອບການພິຈາລະນາກໍໄດ້ ທັງນີ້ເພື່ອໃຫ້ໄດ້ການຈຳແນກທີ່ຖືກຕ້ອງທີ່ສຸດ.

ບົດທີ 6 ເຄືອຂ່າຍເສັ້ນປະສາດທຽມ

ເຄືອຂ່າຍເສັ້ນປະສາດທຽມ ຫຼື ເຄືອຂ່າຍເສັ້ນປະສາດແບບຈຳລອງທາງຄະນິດສາດ ສຳລັບການປະມວນຜົນຂໍ້ມູນດ້ວຍຄອມພິວເຕີທີ່ເນັ້ນໃສ່ການເຊື່ອມຕໍ່ (Connectionist) ເພື່ອຈຳລອງການເຮັດວຽກຂອງເຄືອຂ່າຍເສັ້ນປະສາດໃນສະໝອງມະນຸດ, ຈຸດປະສົງເພື່ອສ້າງເຄື່ອງມືທີ່ມີຄວາມສາມາດໃນການຮຽນຮູ້, ການຈື່ຈຳຮູບແບບ (Pattern Recognition) ແລະ ການຄັດຈ້ອນຄວາມຮູ້ (Knowledge deduction) ຄືກັນກັບຄວາມສາມາດທີ່ມີໃນສະໝອງມະນຸດ.

ແນວຄິດທຳອິດຂອງເຕັກນິກນີ້ໄດ້ມາຈາກການສຶກສາດ້ານເຄືອຂ່າຍໄຟຟ້າຊີວະພາບ (Bioelectric Network) ໃນສະໝອງ, ເຊິ່ງປະກອບດ້ວຍ ເຊວລະບົບປະສາດ (Nerve Cells) ຫຼື “ນິວຣອນ” (neurons) ແລະ ຈຸດເຊື່ອມຕໍ່ປະສາດ (Synapses), ແຕ່ລະເຊວປະສາດປະກອບດ້ວຍເສັ້ນເສັ້ນການຮັບກະແສປະສາດ ເອີ້ນວ່າ "ເດນໄດຣທ໌" (Dendrite) ເຊິ່ງເປັນ Input ແລະ ເສັ້ນເສັ້ນການສົ່ງກະແສປະສາດເອີ້ນວ່າ "ແອກຊອນ" (Axon) ເຊິ່ງເປັນຄື Output ຂອງເຊວ ເຊວເຫຼົ່ານີ້ເຮັດວຽກດ້ວຍການປະຕິກິລິຍາໄຟຟ້າເຄມີ ເມື່ອມີການກະຕຸ້ນຈາກພາຍນອກ ຫຼື ກະຕຸ້ນດ້ວຍເຊວນຳກັນ ກະແສປະສາດຈະແລ່ນຜ່ານເດນໄດຣທ໌ເຂົ້າສູ່ແກ່ນນ້ອຍເຊິ່ງຈະເປັນຕົວຕັດສິນວ່າຕ້ອງກະຕຸ້ນເຊວອື່ນໆຕໍ່ ຫຼືບໍ່? ຖ້າກະແສປະສາດແຮງພໍ ແກ່ນນ້ອຍກໍຈະກະຕຸ້ນເຊວອື່ນໆຕໍ່ໄປຜ່ານທາງແອກຊອນຂອງມັນ. ແບບຈຳລອງເຄືອຂ່າຍປະສາດເກີດຈາກການເຊື່ອມຕໍ່ລະຫວ່າງເຊວລະບົບປະສາດ ຈົນກາຍເປັນເຄືອຂ່າຍທີ່ເຮັດວຽກຮ່ວມກັນໄດ້



ຮູບທີ 6.1 ສະແດງ Model ຂອງ Neuron ໃນສະໝອງມະນຸດ

6.1 ຄວາມໝາຍ

ລະບົບເຄືອຂ່າຍເສັ້ນປະສາດທຽມ (Neural Network) ຫຼື "ເຄືອຂ່າຍເສັ້ນປະສາດທຽມ (Artificial Neural Network: ANN)" ໝາຍເຖິງຄອມພິວເຕີທີ່ສາມາດຮຽນແບບການເຮັດວຽກຂອງສະໝອງມະນຸດ ດ້ວຍການປະມວນຜົນຂໍ້ມູນຂ່າວສານ ແລະ ຄວາມຮູ້ໄດ້ເທື່ອລະຫຼາຍໆ.

ນອກຈາກນີ້, ຍັງສາມາດຮັບ ແລະ ຈື່ຈຳຂ່າວສານໃນຮູບແບບທີ່ເປັນປະສົບການໄດ້, ເຮັດໃຫ້ສາມາດ ເຊື່ອມຕໍ່ຂໍ້ເທັດຈິງທັງຫຼາຍເຂົ້ານຳກັນເພື່ອຫາຂໍ້ສະຫຼຸບ ແລະ ໃຊ້ປະສົບການທີ່ຈັດເກັບໄວ້ມາຮຽນຮູ້ ແລະ ສ້າງຄວາມເຂົ້າໃຈວ່າ ຂໍ້ເທັດຈິງໃໝ່ທີ່ໄດ້ຮັບເຂົ້າມາມີຄວາມກ່ຽວຂ້ອງກັນແນວໃດ ເພື່ອກໍ່ການປັບປຸງ ຄວາມຮູ້ໃຫ້ມີຄວາມທັນສະໄໝເພື່ອປະໂຫຍດໃນອະນາຄົດ.

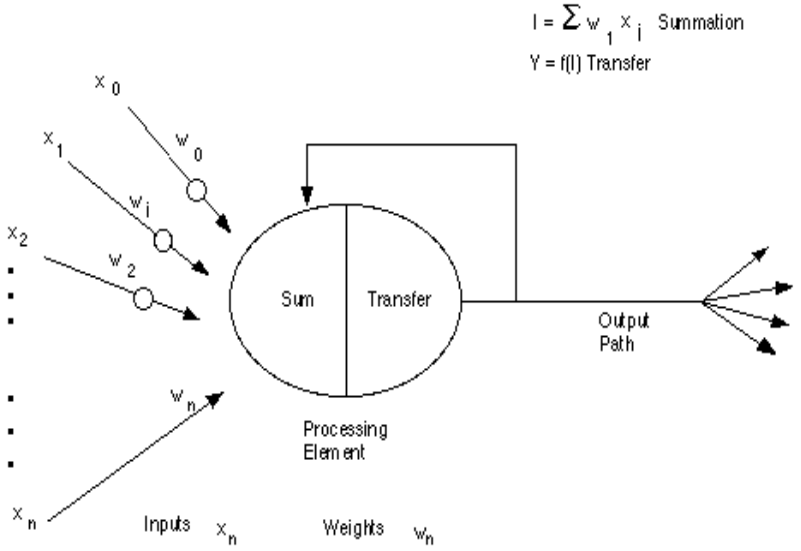
1. Neural Network ເປັນຕົວປະມວນຜົນຄູ່ຂະໜານຂະໜາດໃຫຍ່ ທີ່ສ້າງຂຶ້ນຈາກໜ່ວຍປະມວນຜົນຂະໜາດນ້ອຍ, ມີຄຸນສົມບັດເພື່ອເກັບສິ່ງທີ່ຮັບຮູ້, ປະສົບການ ຫຼື ການເຮັດວຽກ ມີລັກສະນະຄ້າຍຄືກັບ ສະໝອງ 2 ຂໍ້ ຄື:
    - ສິ່ງທີ່ຮັບຮູ້ໄດ້ມາດ້ວຍເຄືອຂ່າຍ (Network) ເຊິ່ງໄດ້ຜ່ານຂະບວນການຮຽນຮູ້
    - ເຊວທີ່ເຊື່ອມຕໍ່ຫາກັນ ເອີ້ນວ່າ Synaptic ຈະຖືກໃຊ້ເພື່ອເກັບສິ່ງທີ່ຮັບຮູ້ເຂົ້າມາ

2.ເຄືອຂ່າຍເສັ້ນປະສາດທຽມ ແລະ ເສັ້ນປະສາດຈິງ

- ເຄືອຂ່າຍເສັ້ນປະສາດທຽມເປັນການຮຽນແບບການເຮັດວຽກຂອງສະໝອງມະນຸດ ທີ່ປະກອບໄປດ້ວຍ ເຊວພິເສດຫຼວງຫຼາຍທີ່ເອີ້ນວ່າ "ເຊວລະບົບປະສາດ (Neuron)" ເຊິ່ງມີຫຼາຍກວ່າ 100 ຊະນິດ.
- ເຊວປະສາດທີ່ມີຊະນິດດຽວກັນຈະຖືກຈັດໄວ້ໃນກຸ່ມດຽວກັນ ເອີ້ນວ່າ "ເຄືອຂ່າຍ (Network)" ແຕ່ລະ ເຄືອຂ່າຍຈະບັນຈຸເຊວປະສາດຈຳນວນນັບ 1000 ເຊວທີ່ມີການເຊື່ອມຕໍ່ກັນຢ່າງໜຽວແໜ້ນ, ດັ່ງນັ້ນ ສະໝອງມະນຸດຈຶ່ງເອີ້ນໄດ້ອີກຢ່າງໜຶ່ງວ່າ "ກຸ່ມປະສາດ".
- ການຮຽນແບບການເຮັດວຽກຂອງສະໝອງມະນຸດຂອງເຄື່ອງຄອມພິວເຕີ ເລີ່ມຈາກການກຳນົດໃຫ້ແຕ່ ລະຊອບແວເອີ້ນວ່າ "ໂນດ (Node)" ທຽມວ່າເປັນ "ເຊວລະບົບປະສາດ"
- ແລະສ້າງການເຊື່ອມຕໍ່ໃຫ້ກັບໂນດເຫຼົ່ານັ້ນໃຫ້ເປັນເຄືອຂ່າຍ (Network), ແຕ່ລະເຄືອຂ່າຍຈະປະກອບ ໄປດ້ວຍໂນດທີ່ຖືກຈັດແບ່ງເປັນຊັ້ນໆ ເອີ້ນວ່າ "ເລເຢີ (Layer)" ແຕ່ລະເລເຢີຈະມີໜ້າທີ່ການເຮັດ ວຽກແຕກຕ່າງກັນ

6.2. ອົງປະກອບ ແລະ ໂຄງສ້າງການເຮັດວຽກ

Software ທີ່ຮຽນແບບເຄືອຂ່າຍຂອງເຊວປະສາດນັ້ນຈະມີຂອບເຂດ (Boundary) ຂັ້ນລະຫວ່າງຂໍ້ມູນນຳເຂົ້າກັບການເຮັດວຽກຂອງເສັ້ນປະສາດທຽມ, ເຊິ່ງປະກອບດ້ວຍເຄືອຂ່າຍຂອງເຊວທີ່ຖືກຈັດໄວ້ເປັນຊັ້ນ, ດັ່ງນັ້ນອົງປະກອບທີ່ໄດ້ຈັດແບ່ງເປັນຊັ້ນ ແລະ ໜ້າທີ່ຂອງແຕ່ລະອົງປະກອບຈຶ່ງມີດັ່ງນີ້



ຮູບທີ 6.2 ອົງປະກອບ ແລະ ໂຄງສ້າງການເຮັດວຽກຂອງເຄືອຂ່າຍເສັ້ນປະສາດທຽມ

6.2.1. ຂໍ້ມູນນຳເຂົ້າ (Input)

ຂໍ້ມູນນຳເຂົ້າຈະຖືກຈຳແນກຕາມຄຸນລັກສະນະ (Attribute) ເຊັ່ນ: ຖ້າບັນຫາທີ່ລະບົບເສັ້ນປະສາດທຽມຈະຕ້ອງຕັດສິນໃຈຄື ຕົວຢ່າງການອະນຸມັດເງິນກູ້ວ່າຈະໃຫ້ຜ່ານ ຫຼືບໍ່? ຂໍ້ມູນນຳເຂົ້າກໍຈະຖືກຈຳແນກເປັນຄຸນລັກສະນະຄື: ລະດັບລາຍຮັບ ແລະ ອາຍຸ ເປັນຕົ້ນ. ຂໍ້ມູນນຳເຂົ້ານອກຈາກຈະເປັນຂໍ້ຄວາມແລ້ວ ຍັງສາມາດເປັນຮູບພາບ ຫຼື ສຽງກໍໄດ້, ແຕ່ອາດຈະຕ້ອງຜ່ານການແປງໃຫ້ເປັນສັນຍະລັກ ຫຼື ຕົວເລກເພື່ອໃຫ້ເຄື່ອງສາມາດສ້າງຄວາມເຂົ້າໃຈໄດ້ກ່ອນ ຈາກນັ້ນກໍຈະເຂົ້າສູ່ການເຮັດວຽກທີ່ແທ້ຈິງຂອງລະບົບເສັ້ນປະສາດທຽມທີ່ເລີ່ມຕົ້ນດ້ວຍການນຳຂໍ້ມູນເຂົ້າມາ, ໃຫ້ນ້ຳໜັກ (Weight) ຂອງຄວາມສຳພັນລະຫວ່າງຂໍ້ມູນນຳເຂົ້າເຫຼົ່ານັ້ນໃນຊັ້ນທຳອິດພາຍໃຕ້ຂອບເຂດຂອງລະບົບ.

6.2.2. ນ້ຳໜັກ (Weight)

ເປັນສ່ວນປະກອບທີ່ສຳຄັນຂອງລະບົບເຄືອຂ່າຍເສັ້ນປະສາດ ເນື່ອງຈາກເປັນສ່ວນທີ່ໃຊ້ຫານ້ຳໜັກຂອງຄວາມສຳພັນລະຫວ່າງຂໍ້ມູນນຳເຂົ້າ, ວ່າຂໍ້ມູນນຳເຂົ້າໃດມີຄວາມສຳພັນກັບຂໍ້ມູນນຳເຂົ້າອື່ນໃນລະດັບໃດ, ເຊິ່ງຈະເຮັດໃຫ້ສາມາດເຊື່ອມຕໍ່ໄປຫາຂໍ້ສະຫຼຸບໄດ້ ດ້ວຍການລອງຜິດລອງຖືກໃນຄວາມສຳພັນແຕ່ລະແບບ ແລະ ເກັບໄວ້ເປັນແບບແຜນ ຫຼື ຮູບແບບ (Pattern) ຂອງປະສົບການເພື່ອການຮຽນຮູ້ຂອງເຄືອຂ່າຍ.

6.2.3. ฟังຊັ້ນການລວມ (Summation Function)

ເປັນເຄືອຂ່າຍທີ່ເຮັດໜ້າທີ່ໃນການລວມຄ່ານ້ຳໜັກທີ່ໄດ້ຈາກເຄືອຂ່າຍໃນຊັ້ນ input ເພື່ອສະຫຼຸບ ຜົນຄວາມສຳພັນລະຫວ່າງຂໍ້ມູນນຳເຂົ້າ ລໍຖ້າການແປງເປັນຂໍ້ມູນຂ່າວສານທີ່ມີຄວາມໝາຍໃນຊັ້ນຕໍ່ໄປ.

6.2.4. ฟังຊັ້ນການแปง (Transformation Function)

ເປັນເຄືອຂ່າຍທີ່ເຮັດໜ້າທີ່ໃນການເຊື່ອມຕໍ່ (Integrate) ຂ່າວສານທີ່ຜ່ານການປະມວນຜົນຈາກ ເຄືອຂ່າຍໃນຊັ້ນຕ່າງໆ, ແລ້ວກໍ່ແປງ (Transform) ໃຫ້ກາຍເປັນຂ່າວສານທີ່ສື່ຄວາມໝາຍ ແລະ ເປັນ ປະໂຫຍດຕໍ່ການເອົາໄປໃຊ້ໄດ້ເພື່ອສົ່ງອອກໄປເປັນຜົນຮັບ (Output).

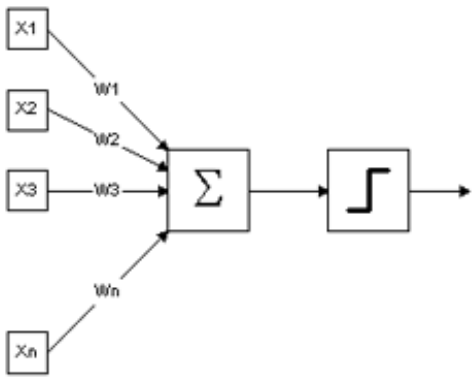
6.2.5. ผົนรับ (Output)

ຜົນຮັບທີ່ໄດ້ຈາກເຄືອຂ່າຍເສັ້ນປະສາດທຽມ ຈະໝາຍເຖິງແນວທາງໃນການແກ້ໄຂບັນຫາ, ເຊັ່ນ: ບັນຫາການອະນຸມັດເງິນກູ້ວ່າຜູ້ກູ້ຈະຜ່ານການອະນຸມັດ ຫຼືບໍ່? “ຜົນຮັບ” ທີ່ຜູ້ໃຊ້ຈະໄດ້ຮັບຄື ”ອະນຸມັດ” ຫຼື “ບໍ່ອະນຸມັດ”, ເຊິ່ງເຄືອຂ່າຍເສັ້ນປະສາດທຽມຈະໃຊ້ສັນຍະລັກແທນຄຳຕອບທັງໝົດ.

ຜົນຮັບທີ່ໄດ້ຈາກເຄືອຂ່າຍໜຶ່ງສາມາດເປັນຂໍ້ມູນນຳເຂົ້າ (Input) ຂອງເຄືອຂ່າຍໜຶ່ງໄດ້, ໃນນີ້ເພື່ອ ເປັນຂໍ້ມູນນຳເຂົ້າຂອງການຕັດສິນໃຈແກ້ໄຂບັນຫາອື່ນ ເຊັ່ນ: ຜົນຮັບທີ່ໄດ້ຈາກການອະນຸມັດເງິນກູ້ ອາດ ຈະນຳໄປໃຊ້ເປັນຂໍ້ມູນນຳເຂົ້າເພື່ອການອະນຸມັດສິນເຊື່ອທີ່ຢູ່ອາໄສໄດ້.

6.3. ໂຄງสร้าง

ນັກວິໄຈສ່ວນຫຼາຍໃນປັດຈຸບັນເຫັນກົງກັນວ່າເຄືອຂ່າຍເສັ້ນປະສາດທຽມມີໂຄງສ້າງແຕກຕ່າງຈາກ ເຄືອຂ່າຍເສັ້ນສະໝອງມະນຸດ ແຕ່ກໍຍັງຄືສະໝອງ ໃນທາງທີ່ວ່າເຄືອຂ່າຍເສັ້ນປະສາດທຽມ ຄືການລວມ ກຸ່ມແບບຂະໜານຂອງໜ່ວຍປະມວນຜົນຍ່ອຍໆ ແລະ ການເຊື່ອມຕໍ່ນີ້ເປັນສ່ວນສຳຄັນທີ່ເຮັດໃຫ້ເກີດສະຕິ ປັນຍາຂອງເຄືອຂ່າຍ. ເມື່ອພິຈາລະນາຂະໜາດແລ້ວ ສະໝອງມະນຸດມີຂະໜາດໃຫຍ່ກວ່າເຄືອຂ່າຍ ເສັ້ນປະສາດທຽມຫຼາຍ ລວມທັງເຊວປະສາດຍັງມີຄວາມຊັບຊ້ອນກວ່າໜ່ວຍຍ່ອຍຂອງເຄືອຂ່າຍ ຢ່າງໃດກໍ ດີໜ້າທີ່ສຳຄັນຂອງສະໝອງ ເຊັ່ນການຮຽນຮູ້ຍັງສາມາດຖືກຈຳລອງຂຶ້ນຢ່າງງ່າຍດ້ວຍເຄືອຂ່າຍ ເສັ້ນປະສາດທຽມນີ້.



ຮູບທີ 6.3 ສະແດງ Model ຂອງ Neuron ໃນຄອມພິວເຕີ

6.4. ຫຼັກການ

ສຳລັບໃນຄອມພິວເຕີ Neurons ປະກອບດ້ວຍ input ແລະ output ຄືກັນ ໂດຍຈຳລອງໃຫ້ input ແຕ່ລະອັນມີ weight ເປັນຕົວກຳນົດນ້ຳໜັກຂອງ input ໂດຍ neuron ແຕ່ລະໜ່ວຍຈະມີຄ່າ threshold ເປັນຕົວກຳນົດວ່ານ້ຳໜັກລວມຂອງ input ຕ້ອງຫຼາຍຂະໜາດໃດຈຶ່ງຈະສາມາດສົ່ງ output ໄປ ທີ່ neurons ຕົວອື່ນໄດ້. ເມື່ອເອົາ neuron ແຕ່ລະໜ່ວຍມາຕໍ່ກັນໃຫ້ເຮັດວຽກຮ່ວມກັນ ການເຮັດວຽກນີ້ໃນ ທາງຕັກກະສາດແລ້ວກໍຈະຄືກັບປະຕິກິລິຍາເຄມີທີ່ເກີດໃນສະໝອງມະນຸດ ພຽງແຕ່ໃນຄອມພິວເຕີທຸກຢ່າງ ເປັນຕົວເລກເທົ່ານັ້ນ.

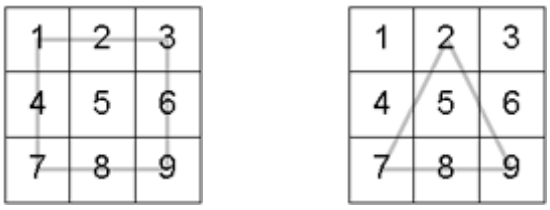
6.5. ການເຮັດວຽກ

ການເຮັດວຽກຂອງ Neural Networks ແມ່ນເມື່ອມີ input ເຂົ້າມາທີ່ network, network ກໍ ເອົາ input ມາຄູນກັບ weight ຂອງແຕ່ລະຂາ, ຜົນທີ່ໄດ້ຈາກ input ທຸກໆຂາຂອງ neuron ຈະເອົາ ມາລວມກັນແລ້ວກໍເອົາມາທຽບກັບ threshold ທີ່ກຳນົດໄວ້. ຖ້າຜົນທັງໝົດມີຄ່າຫຼາຍກວ່າ threshold ແລ້ວ neuron ກໍຈະສົ່ງ output ອອກໄປ, output ນີ້ກໍຈະຖືກສົ່ງໄປຍັງ input ຂອງ neuron ອື່ນໆ ທີ່ເຊື່ອມກັນ ໃນ network. ຖ້າຄ່ານ້ອຍກວ່າ threshold ກໍຈະບໍ່ເກີດ output. ການເຮັດວຽກຂອງ neural ສາມາດຂຽນ ອອກມາໄດ້ດັ່ງນີ້:

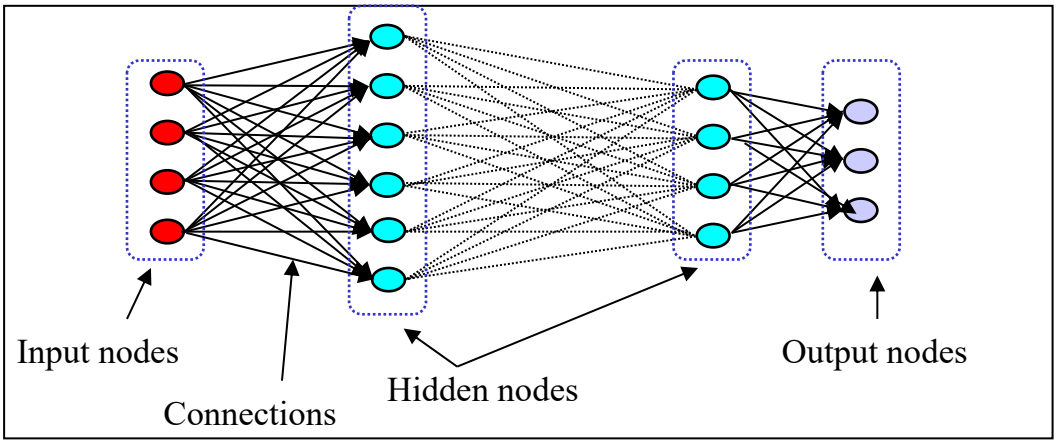
if (sum(input * weight) > threshold) then output

ສິ່ງສຳຄັນຄືເຮົາຕ້ອງຮູ້ຄ່າ weight ແລະ threshold ສຳລັບສິ່ງທີ່ເຮົາຕ້ອງການເພື່ອໃຫ້ຄອມພິວ ເຕີຮູ້, ເຊິ່ງເປັນຄ່າທີ່ບໍ່ແນ່ນອນ ແຕ່ສາມາດກຳນົດໃຫ້ຄອມພິວເຕີປັບຄ່າເຫຼົ່ານັ້ນໄດ້ ໂດຍການສອນໃຫ້ມັນ ຮູ້ຈັກ pattern ຂອງສິ່ງທີ່ເຮົາຕ້ອງການໃຫ້ມັນຈື່ຈຳ ເອີ້ນວ່າ "back propagation" ເຊິ່ງເປັນຂະບວນການ ກັບຄືນຂອງການຈື່ຈຳ. ໃນການຝຶກ feed-forward neural networks ຈະມີການໃຊ້ອານຸກໍຣິດທຶມແບບ back-propagation ເພື່ອໃຊ້ໃນການປັບປຸງນ້ຳໜັກຄະແນນຂອງເຄືອຂ່າຍ (network weight), ຫຼັງຈາກໃສ່ ຮູບແບບຂໍ້ມູນສຳລັບຝຶກໃຫ້ແກ່ເຄືອຂ່າຍໃນແຕ່ລະເທື່ອແລ້ວ ຄ່າທີ່ໄດ້ຮັບ (output) ຈາກເຄືອຂ່າຍຈະຖືກ ນຳໄປປຽບທຽບກັບຜົນທີ່ຄາດຫວັງ, ແລ້ວກໍ່ສ້າງການຄິດໄລ່ຫາຄ່າຄວາມຜິດພາດ, ເຊິ່ງຄ່າຄວາມຜິດພາດ ນີ້ຈະຖືກສົ່ງກັບເຂົ້າສູ່ເຄືອຂ່າຍເພື່ອໃຊ້ແກ້ໄຂຄ່ານ້ຳໜັກຄະແນນຕໍ່ໄປ.

ຕົວຢ່າງເຊັ່ນຈະຈື່ຈຳຮູບສາມຫຼ່ຽມ ກັບຮູບສີ່ຫຼ່ຽມ ເຮົາອາດແບ່ງ input ເປັນ 9 ຕົວຄືເປັນຕາ ຕະລາງ 3x3 ຖ້າແຕ້ມຮູບສີ່ຫຼ່ຽມ ຫຼື ສາມຫຼ່ຽມໃຫ້ເຕັມຂອບ 3x3 ພໍດີ, ສີ່ຫຼ່ຽມຈະມີສ່ວນຂອງຂອບຢູ່ໃນ ປ່ອງ 1,2,3,4,6,7,8,9 ສົມມຸດໃຫ້ນ້ຳໜັກບ່ອນປ່ອງເຫຼົ່ານີ້ມີຄ່າຫຼາຍໆ ຖ້າມີເສັ້ນຂີດຜ່ານກໍເອົາມາຄູນກັບນ້ຳ ໜັກແລ້ວກໍເອົາມາລວມກັນ ຕັ້ງຄ່າໃຫ້ພໍດີ ກໍຈະສາມາດແຍກລະຫວ່າງສີ່ຫຼ່ຽມກັບສາມຫຼ່ຽມໄດ້, ເຊິ່ງນີ້ຄື ຫຼັກການຂອງ **neural network**



ຮູບທີ 6.4 ສະແດງການແຍກລະຫວ່າງສີ່ຫຼ່ຽມ ແລະ ສາມຫຼ່ຽມ



ຮູບທີ 6.5 ສະແດງໂຄງສ້າງວົງຈອນເຄືອ Neural Network

❖ **Output ຂອງແຕ່ລະ Node**

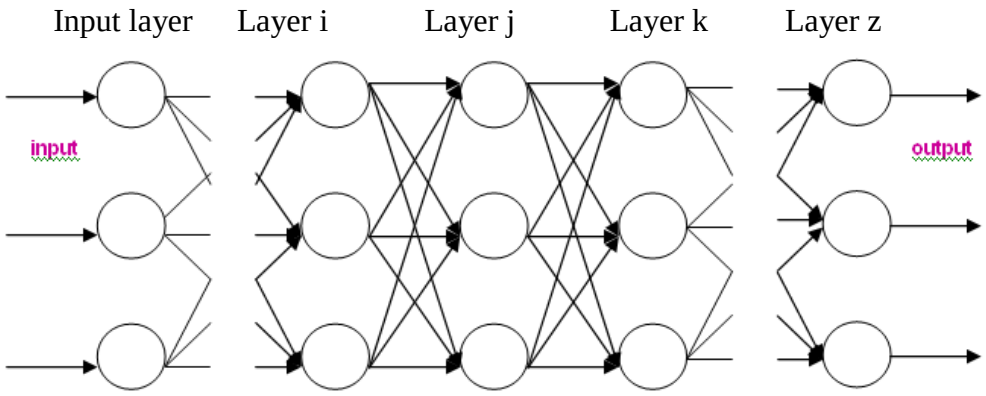
$$y_i = f(w_i^1 x_1 + w_i^2 x_2 + w_i^3 x_3 + \cdots + w_i^m x_m) = f(\sum_j w_i^j x_j)$$

ເມື່ອ *Xi* = input ຈາກໂນດອື່ນໆ

Wij = ນ້ຳໜັກ (weight) ຂອງແຕ່ລະແຂນ (connection)

6.5.1. Back propagation Algorithm

Back-propagation ເປັນອານກໍຣິດທຶມທີ່ໃຊ້ໃນການຮຽນຮູ້ຂອງເຄືອຂ່າຍເສັ້ນປະສາດວິທີໜຶ່ງທີ່ນິຍົມໃຊ້ໃນ Multilayer Perceptron ເພື່ອປັບຄ່ານ້ຳໜັກໃນເສັ້ນເຊື່ອມຕໍ່ລະຫວ່າງໂນດໃຫ້ເໝາະສົມ ໂດຍການປັບຄ່ານີ້ຈະຂຶ້ນກັບຄວາມແຕກຕ່າງຂອງຄ່ານຳເຂົ້າທີ່ຄິດໄລ່ໄດ້ກັບຄ່ານຳເຂົ້າທີ່ຕ້ອງການ ພິຈາລະນາຮູບຕໍ່ໄປນີ້.



ຮູບທີ 6.6 ສະແດງຮູບປະກອບ Back-propagation Neuron Network

❖ **ຂັ້ນຕອນຂອງ Back-propagation Algorithm ມີດັ່ງນີ້**

1. ກຳນົດຄ່າອັດຕາຄວາມໄວໃນການຮຽນຮູ້ (Rate Parameter : r)

2. ສຳລັບແຕ່ລະຕົວຢ່າງ input ໃຫ້ເຮັດຕາມຂັ້ນຕອນຕໍ່ໄປນີ້ຈົນກວ່າໄດ້ລະດັບ performance ທີ່ຕ້ອງການ

- ຄິດໄລ່ຫາຄ່ານຳເຂົ້າໂດຍໃຊ້ຄ່ານ້ຳໜັກເລີ່ມຕົ້ນເຊິ່ງອາດໄດ້ຈາກການສຸ່ມ
- ຄິດໄລ່ຫາຄ່າ (ແທນປະໂຫຍດທີ່ຈະໄດ້ຮັບສຳລັບການປ່ຽນຄ່ານຳເຂົ້າຂອງແຕ່ລະໂນດ)
- ໃນຊັ້ນນຳເຂົ້າ **(Output Layer)**

$$\beta_z = d_z - o_z$$

ເມື່ອ d z ແມ່ນຄ່ານຳເຂົ້າທີ່ຕ້ອງການ

Oz ແມ່ນຄ່ານຳເຂົ້າທີ່ຄິດໄລ່ໄດ້

- ໃນຊັ້ນເຊື່ອງ (Hidden Layer)

$$\beta_j = \sum w_{jk} * o_k * (1 - o_k) * \beta_k$$

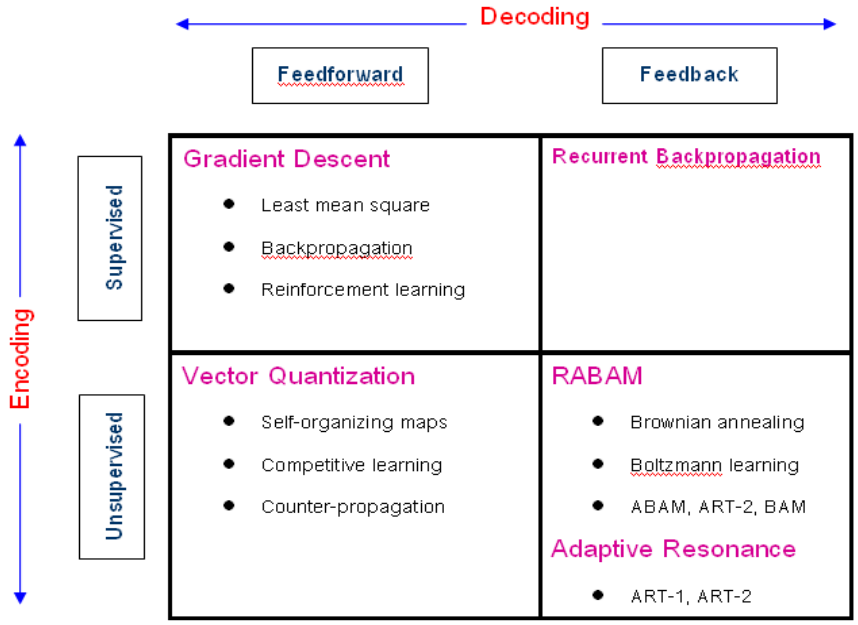
ເມື່ອ W_{jk} ແມ່ນນ້ຳໜັກຂອງເສັ້ນເຊື່ອມລະຫວ່າງຊັ້ນທີ j ກັບ k

- ຄິດໄລ່ຄ່ານ້ຳໜັກທີ່ປ່ຽນແປງໄປສຳລັບໃນທຸກນ້ຳໜັກ ດ້ວຍສົມຜົນຕໍ່ໄປນີ້

$$\Delta w_{ij} = r * o_i * o_j * (1 - o_j) * \beta_j$$

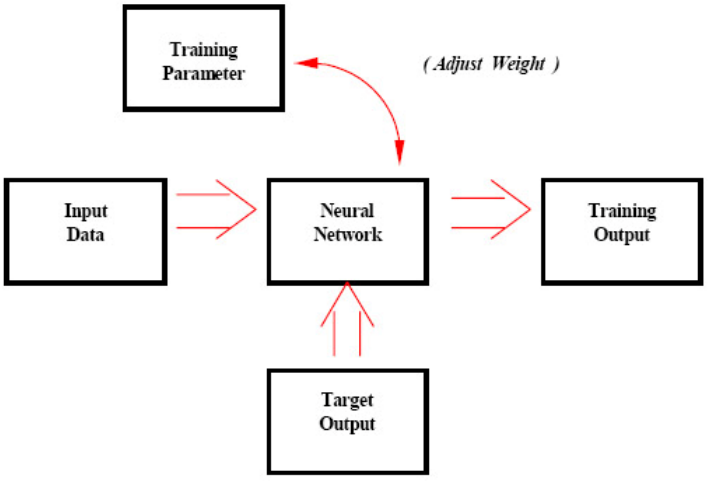
- ເພີ່ມຄ່ານ້ຳໜັກທີ່ປ່ຽນແປງ ສຳລັບຕົວຢ່າງ input ທັງໝົດ ແລະ ປ່ຽນຄ່ານ້ຳໜັກ

6.5.2. ການຮຽນຮູ້ສຳລັບ Neural Network



1. ການຮຽນຮູ້ແບບມີການສອນ (Supervised Learning)

ເປັນການຮຽນແບບທີ່ມີການກວດຄຳຕອບເພື່ອໃຫ້ວົງຈອນເຄືອຂ່າຍປັບຕົວ, ຊຸດຂໍ້ມູນທີ່ໃຊ້ສອນວົງຈອນເຄືອຂ່າຍຈະມີຄຳຕອບໄວ້ຖ້າກວດເບິ່ງວ່າວົງຈອນເຄືອຂ່າຍໃຫ້ຄຳຕອບທີ່ຖືກ ຫຼືບໍ່? ຖ້າຕອບບໍ່ຖືກ ວົງຈອນເຄືອຂ່າຍກໍຈະປັບຕົວເອງເພື່ອໃຫ້ໄດ້ຄຳຕອບທີ່ດີຂຶ້ນ (ປຽບທຽບກັບຄົນຄືກັບການສອນນັກຮຽນໂດຍມີຄູຜູ້ສອນຄອຍຖ້າແນະນຳ).

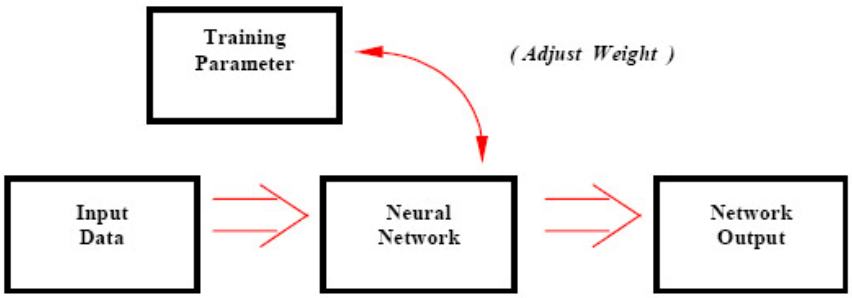


ຮູບທີ 6.7 ສະແດງການຮຽນຮູ້ແບບມີການສອນ (Supervised Learning)

2. ການຮຽນແບບບໍ່ມີການສອນ (Unsupervised Learning)

ເປັນການຮຽນແບບບໍ່ມີຜູ້ແນະນຳ, ບໍ່ມີການກວດຄຳຕອບວ່າຖືກ ຫຼືຜິດ, ວົງຈອນເຄືອຂ່າຍຈະຈັດລຽງໂຄງສ້າງດ້ວຍຕົວເອງຕາມລັກສະນະຂອງຂໍ້ມູນ ຜົນຮັບທີ່ໄດ້ວົງຈອນເຄືອຂ່າຍຈະສາມາດຈັດ

ໝວດໝູ່ຂອງຂໍ້ມູນໄດ້ (ປຽບທຽບກັບຄົນເຊັ່ນ: ການທີ່ເຮົາສາມາດແຍກພັນພືດ, ພັນສັດຕາມລັກສະນະຮູບຮ່າງຂອງມັນໄດ້ເອງໂດຍບໍ່ມີໃຜສອນ).

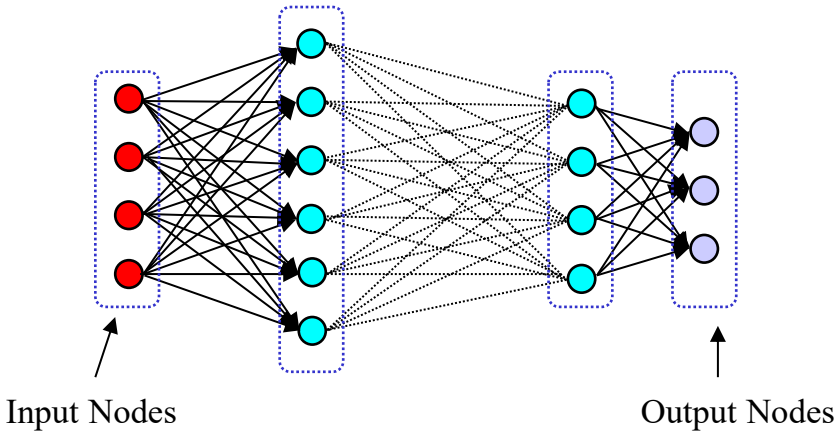


ຮູບທີ 6.8 ສະແດງການຮຽນຮູ້ແບບບໍ່ມີການສອນ Unsupervised Learning

6.6. Network Architecture

6.6.1. Feedforward network

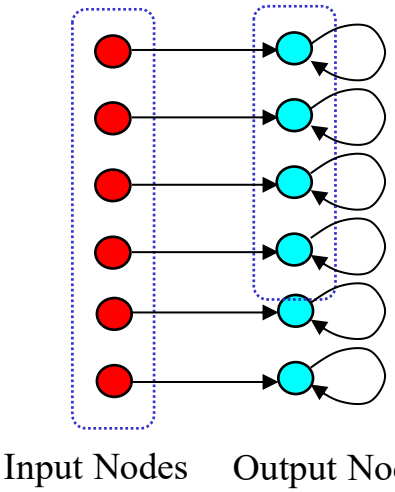
ຂໍ້ມູນທີ່ປະມວນຜົນໃນວົງຈອນເຄືອຂ່າຍຈະຖືກສົ່ງໄປໃນທິດທາງດຽວຈາກ Input Nodes ສົ່ງຕໍ່ມາເລື້ອຍໆ ຈົນເຖິງ Output Nodes ໂດຍບໍ່ມີການກັບຄືນຂອງຂໍ້ມູນ ຫຼື ລວມທັງ Nodes ໃນ layer ດຽວກັນກໍບໍ່ມີການເຊື່ອມຕໍ່ກັນ.



ຮູບທີ 6.9 ສະແດງສະຖາປັດຕະຍະກຳຂອງ Feedforward network

6.6.2. Feedback network

ຂໍ້ມູນທີ່ປະມວນຜົນໃນວົງຈອນເຄືອຂ່າຍ ຈະມີການປ້ອນກັບເຂົ້າໄປທີ່ວົງຈອນເຄືອຂ່າຍຫຼາຍໆເທື່ອ ຈົນກວ່າໄດ້ຄຳຕອບອອກມາ (ບາງຄັ້ງເອີ້ນວ່າ Recurrent network)



ຮູບທີ 6.10 ສະແດງສະຖາປັດຕະຍະກຳຂອງ Feedback network

6.6.3. Network Layer

ພື້ນຖານທີ່ສຳຄັນຂອງ Artificial Neural Network ປະກອບໄປດ້ວຍ 3 ສ່ວນ ຫຼື 3 layer ໄດ້ແກ່ ຊັ້ນຂອງ input units ທີ່ຖືກເຊື່ອມຕໍ່ກັບຊັ້ນຂອງ hidden units ເຊິ່ງເຊື່ອມຕໍ່ກັບຊັ້ນຂອງ output units

- ການເຮັດວຽກຂອງ input unit ຈະເຮັດໜ້າທີ່ແທນສ່ວນຂອງຂໍ້ມູນດິບ ທີ່ຈະຖືກປ້ອນເຂົ້າສູ່ເຄືອຂ່າຍ
- ການເຮັດວຽກຂອງແຕ່ລະ hidden units ຈະຖືກກຳນົດ ໂດຍການເຮັດວຽກຂອງ input units ແລະ ຄ່ານ້ຳໜັກເທິງຄວາມສຳພັນລະຫວ່າງ input units ແລະ hidden units
- ພຶດຕິກຳການເຮັດວຽກຂອງ output units ຈະຂຶ້ນຢູ່ກັບການເຮັດວຽກຂອງ hidden units ແລະ ຄ່ານ້ຳໜັກລະຫວ່າງ hidden units ແລະ output units

❖ **Architecture of Layer**

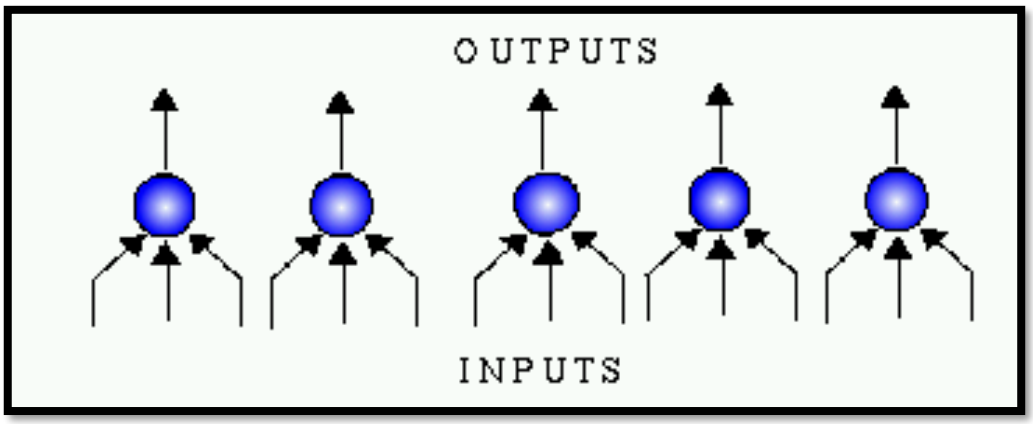
ສາມາດຈຳແນກສະຖາປັດຕະຍະກຳຂອງຊັ້ນ (layer) ອອກເປັນ 2 ປະເພດຄື Single-layer ແລະ Multi-layer

- **Single-layer perceptron** ເຄືອຂ່າຍເສັ້ນປະສາດທີ່ປະກອບດ້ວຍຊັ້ນໜຶ່ງຊັ້ນດຽວ ຈຳນວນ input nodes ຂຶ້ນຢູ່ກັບຈຳນວນ components ຂອງ input data ແລະ Activation Function ຂຶ້ນຢູ່ກັບລັກສະນະຂໍ້ມູນຂອງ Output ເຊັ່ນ: ຖ້າ output ທີ່ຕ້ອງການເປັນ “ແມ່ນ” ຫຼື “ບໍ່ແມ່ນ” ເຮົາຈະຕ້ອງໃຊ້ Threshold function

$$f(x) = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

T ແມ່ນThreshold level

ຫຼືຖ້າ output ເປັນຄ່າຕົວເລກທີ່ຕໍ່ເນື່ອງ ເຮົາຕ້ອງໃຊ້ continuous function ເຊັ່ນ Sigmoid function $f(x) = \frac{1}{1+e^{-\alpha x}}$



ຮູບທີ 6.11 ສະແດງ Single-layer perceptron

- **Multi-layer perceptron** ເຄືອຂ່າຍເສັ້ນປະສາດຈະປະກອບດ້ວຍຫຼາຍຊັ້ນໂດຍໃນແຕ່ລະຊັ້ນຈະປະກອບດ້ວຍໂນດ (nodes) ຫຼື ທຽບໄດ້ກັບຕົວເຊວລະບົບປະສາດ(neurons) ຄ່ານ້ຳໜັກຂອງເສັ້ນທີ່ເຊື່ອມຕໍ່ລະຫວ່າງໂນດຂອງແຕ່ລະຊັ້ນ (ມາຕຣິດ W), ຄ່າ bias vector (b) ແລະ ຄ່າ output vector (a) ໂດຍ m ເປັນຕົວເລກບອກລຳດັບຊັ້ນກຳກັບໄວ້ດ້ານເທິງ ເມື່ອ p ເປັນ input vector ການຄິດໄລ່ຄ່ານຳເຂົ້າສຳລັບເຄືອຂ່າຍເສັ້ນປະສາດທີ່ມີ M ຊັ້ນຈະເປັນດັ່ງສົມຜົນ

$$a^{m+1} = f^{m+1}(W^{m+1} a^{m} + b^{m+1})$$

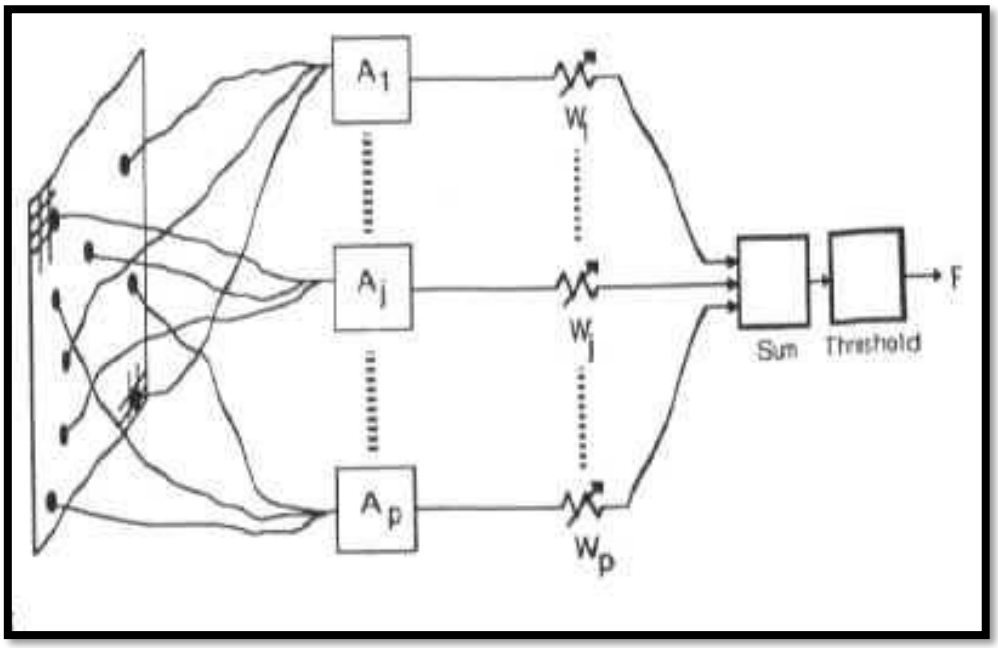
ເມື່ອ m ມີຄ່າ 0, 2, …, M-1

a^0 ແມ່ນ p

a ແມ່ນ a^m ແລະ f ເປັນ transfer function

6.6.4. Perceptrons

ໃນຍຸກສັດຕະວັດ 60 ວົງການສ່ວນໃຫຍ່ຂອງເຄືອຂ່າຍໄດ້ຮັບການວິພາກວິຈານ ໃນຫົວຂໍ້ເລື່ອງ Perceptrons ເຊິ່ງຄົ້ນພົບໂດຍ Frank Rosenblatt, perceptron ກາຍເປັນ MCP model (Neuron with Weighted Inputs) ພ້ອມກັບສ່ວນເພີ່ມເຕີມ. ຈາກຮູບທີ 6.12 ໃນສ່ວນຂອງ A1, A2, Aj, Ap ເອີ້ນວ່າ association units ການເຮັດວຽກເພື່ອຄັດເລືອກສິ່ງທີ່ແຕກຕ່າງອອກມາຈາກຮູບພາບທີ່ຮັບເຂົ້າໄປໂດຍ perceptrons ສາມາດສຳເນົາຄວາມຄິດພື້ນຖານພາຍໃນຂອງສັດລ້ຽງລູກດ້ວຍນົມ ຫຼັກໆແລ້ວຈະໃຊ້ໃນຮູບແບບ recognition ແລະ ສາມາດຂະຫຍາຍໃຫ້ມີຄວາມສາມາດສູງກວ່ານີ້.



ຮູບທີ 6.12 ສະແດງໂຄງສ້າງຂອງ Perceptrons

6.7. ປະໂຫຍດຂອງເຄືອຂ່າຍເສັ້ນປະສາດທຽມ

1. ເກີດຂໍ້ຜິດພາດໄດ້ຍາກ (Fault Tolerance)

ຫາກລະບົບເຄືອຂ່າຍເສັ້ນປະສາດທຽມປະກອບໄປດ້ວຍເຄືອຂ່າຍທີ່ໃຊ້ໃນການປະມວນຜົນຫຼາກຫຼາຍເຄືອຂ່າຍ, ຄວາມຜິດພາດທີ່ເກີດຂຶ້ນຈາກພຽງໜຶ່ງ ຫຼືສອງເຄືອຂ່າຍຈະບໍ່ເຮັດໃຫ້ລະບົບທັງໝົດເກີດຂໍ້ຜິດພາດໄດ້.

2. ຄວາມສາມາດໃນການຫາເຫດຜົນ (Generalization)

ເມື່ອລະບົບເຄືອຂ່າຍເສັ້ນປະສາດທຽມໄດ້ຮັບຂໍ້ມູນນຳເຂົ້າທີ່ບໍ່ຄົບຖ້ວນ ຫຼື ບໍ່ພຽງພໍຕໍ່ການຫາຂໍ້ສະຫຼຸບ, ຫຼື ໄດ້ຮັບຂໍ້ເທັດຈິງທີ່ບໍ່ເຄີຍໄດ້ຮັບມາກ່ອນ, ລະບົບຈະສາມາດລຳດັບການເຊື່ອມຕໍ່ຂໍ້ເທັດຈິງຈົນສາມາດໃຫ້ຂໍ້ສະຫຼຸບ ແລະ ເຫດຜົນໄດ້.

3. ຄວາມສາມາດໃນການປັບປ່ຽນ (Adaptability)

ເຄືອຂ່າຍເສັ້ນປະສາດທຽມສາມາດຮຽນຮູ້ສະພາບແວດລ້ອມໃໝ່ໄດ້, ດັ່ງນັ້ນເມື່ອມີເຫດການໃໝ່ໆເຂົ້າສູ່ລະບົບ, ລະບົບກໍຈະສາມາດປັບປ່ຽນ ຫຼື ປັບປຸງຄວາມຮູ້ໃຫ້ທັນສະໄໝຕາມເຫດການໃໝ່ນັ້ນ.

4. ຄວາມສາມາດໃນການພະຍາກອນ (Forecasting Capability)

ເຄືອຂ່າຍເສັ້ນປະສາດທຽມສາມາດນຳຂໍ້ມູນທາງສະຖິຕິເກົ່າທີ່ມີຢູ່ໃນລະບົບເອົາມາໃຊ້ໃນການຄາດຄະເນ ຫຼື ພະຍາກອນຂໍ້ມູນໃນອະນາຄົດໄດ້.

6.8 ການປະຍຸກໃຊ້ Neural Network

ແບບເຄືອຂ່າຍລະບົບປະສາດ (Neural Network) ເນື່ອງຈາກຄວາມສາມາດໃນການຈຳລອງພຶດຕິກຳທາງກາຍະພາບຂອງລະບົບທີ່ມີຄວາມຊັບຊ້ອນຈາກຂໍ້ມູນທີ່ປ້ອນໃຫ້ຮຽນຮູ້, ການປະຍຸກໃຊ້ເຄືອ

ຂ່າຍລະບົບປະສາດຈຶ່ງເປັນທາງເລືອກໃໝ່ໃນການຄວບຄຸມ, ເຊິ່ງມີຜູ້ນຳມາປະຍຸກໃຊ້ວຽກຫຼາຍປະເພດ ໄດ້ແກ່:

1. ວຽກການຈື່ຈຳຮູບແບບທີ່ມີຄວາມບໍ່ແນ່ນອນ ເຊັ່ນ: ລາຍມື, ລາຍເຊັນ, ຕົວອັກສອນ ແລະ ຮູບໜ້າ
2. ວຽກການປະມານຄ່າຟັງຊັນ ຫຼື ການປະມານຄວາມສຳພັນ (ມີ inputs ແລະ outputs ແຕ່ບໍ່ຮູ້ວ່າ inputs ກັບ outputs ມີຄວາມສຳພັນກັນຢ່າງໃດ).
3. ວຽກທີ່ມີສິ່ງແວດລ້ອມປ່ຽນແປງຢູ່ສະເໝີ (ວົງຈອນເຄືອຂ່າຍນິວຣອນສາມາດປັບຕົວເອງໄດ້).
4. ວຽກຈັດໝວດໝູ່ ແລະ ແຍກສິ່ງຂອງ.
5. ວຽກພະຍາກອນ ເຊັ່ນ: ພະຍາກອນອາກາດ, ພະຍາກອນຫຸ້ນ...
6. ການປະຍຸກໃຊ້ເຄືອຂ່າຍລະບົບປະສາດຄວບຄຸມຂະບວນການທາງເຄມີໂດຍວິທີພະຍາກອນແບບຈຳລອງ (Model Predictive Control)
7. ການປະຍຸກໃຊ້ເຄືອຂ່າຍລະບົບປະສາດແບບແພ່ກະຈາຍກັບ ໃນການພະຍາກອນພະລັງງານຄວາມຮ້ອນທີ່ສະສົມຢູ່ໃນຕຶກອາຄານ
8. ການໃຊ້ເຄືອຂ່າຍລະບົບປະສາດໃນການຫາ psychometric chart, ການປະຍຸກໃຊ້ເຄືອຂ່າຍລະບົບປະສາດຄວບຄຸມລະບົບ HVAC.

6.8.1 ການປະຍຸກໃຊ້ເຄືອຂ່າຍເສັ້ນປະສາດທຽມໃນວຽກທຸລະກິດ

- ການເຮັດເໝືອງຂໍ້ມູນ (Data Mining) ເປັນການເພີ່ມຄວາມສາມາດໃນການຄົ້ນຫາຂໍ້ມູນໃນຖານຂໍ້ມູນຕ່າງຊະນິດ ຫຼື ຖານຂໍ້ມູນທີ່ມີຂະໜາດໃຫຍ່ ແລະ ຊັບຊ້ອນໄດ້.
- ການປ້ອງກັນການໂກງພາສີ (Tax Fraud) ຊ່ວຍລະບຸ ແລະ ຄົ້ນຫາການເຮັດວຽກທີ່ຜິດກົດໝາຍໃນດ້ານການເສຍພາສີໄດ້.
- ການບໍລິການທາງດ້ານການເງິນ (Financial Service) ຊ່ວຍພັດທະນາຮູບແບບການບໍລິການທາງດ້ານການເງິນ ເຊັ່ນ: ການໃຫ້ຂໍ້ມູນຕະຫຼາດຫຸ້ນ ແລະ ເປັນຜູ້ຊ່ວຍການຄ້າຫຸ້ນ ເປັນຕົ້ນ.
- ການວິເຄາະຜະລິດຕະພັນໃໝ່ (New Product Analysis) ຊ່ວຍພະຍາກອນຍອດຂາຍ ແລະ ເລືອກຕະຫຼາດກຸ່ມເປົ້າໝາຍໄດ້.
- ການຈັດການຄ່າທຳນຽມສາຍການບິນ (Airline Fare Management) ຊ່ວຍພະຍາກອນປະລິມານຄວາມຕ້ອງການໃນການຈອງປີ້ຍົນ ແລະ ຈັດຕາຕະລາງກຳລັງຄົນໄດ້.
- ການປະເມີນຜົນ ແລະ ຄັດເລືອກພະນັກງານໃໝ່ ຊ່ວຍຄັດເລືອກພະນັກງານໃໝ່ທີ່ມີຄຸນສົມບັດຕາມທີ່ອົງກອນຕ້ອງການໄດ້.
- ຈັດສັນຊັບພະຍາກອນໃນອົງກອນໂດຍອາໄສຂໍ້ມູນໃນອາດີດ, ເຄືອຂ່າຍເສັ້ນປະສາດທຽມຈະຊ່ວຍສັນຊັບພະຍາກອນທັງໝົດໃນອົງກອນ, ໂດຍອາໄສຂໍ້ມູນໃນອາດີດ ແລະ ທົດລອງປ່ຽນຄ່າຂໍ້ມູນເພື່ອໃຫ້ໄດ້ຜົນຕອບແທນສູງສຸດ.
- ກວດສອບລາຍເຊັນ (Signature Validation) ຊ່ວຍໃນການກວດສອບລາຍເຊັນຈິງກັບລາຍເຊັນທີ່ຈັດເກັບໄວ້ໃນແຟ້ມຂໍ້ມູນ

6.8.2. ຕົວຢ່າງການເຮັດວຽກຂອງເຄືອຂ່າຍເສັ້ນປະສາດ

ເຕັກໂນໂລຢີເຄືອຂ່າຍເສັ້ນປະສາດຈັດວ່າເປັນເຕັກໂນໂລຢີທີ່ມີຄວາມສາມາດສູງ ຈຶ່ງໄດ້ມີການນຳໄປປະຍຸກໃຊ້ກັບລະບົບອື່ນໆເພື່ອປະໂຫຍດໃນການເຮັດວຽກຫຼາຍດ້ານ ຫຼື ມີການນຳໄປເຊື່ອມຕໍ່ເຂົ້າກັບເຕັກໂນໂລຊີອື່ນເພື່ອເພີ່ມຄວາມສາມາດໃຫ້ທຽບເທົ່າກັບມະນຸດ, ດັ່ງຕົວຢ່າງ

1. Synface ການຊ່ວຍເຫຼືອການສົນທະນາທາງໂທລະສັບດ້ວຍໃບໜ້າຈຳລອງ

- ເປັນຊອບແວທີ່ສາມາດສ້າງໃບໜ້າຈຳລອງທີ່ສຳພັນກັບການສົນທະນາຂອງຜູ້ທີ່ຢູ່ປາຍສາຍໂທລະສັບ, ເພື່ອຊ່ວຍເຫຼືອຜູ້ມີບັນຫາທາງການໄດ້ຍິນໄດ້. ພາບໃບໜ້າຈຳລອງເຊິ່ງໃຫ້ພາບຄ້າຍໃບໜ້າຈິງຂອງບຸກຄົນທີ່ກຳລັງສົນທະນາຢູ່ນຳ, ເຮັດໃຫ້ຜູ້ເຫັນສາມາດເຂົ້າໃຈບົດສົນທະນາຈາກການອ່ານຮິບສົບໄດ້ເປັນຢ່າງດີ
- ຊິນເຟດ ໄດ້ຮັບການທົດສອບທີ່ສະຖາບັນຄົນຫູໜວກໃນປະເທດອັງກິດ UK's Royal National Institute for the Deaf (RNID) ພົບວ່າ 84 % ຂອງຜູ້ທີ່ໄດ້ຮັບການທົດສອບສາມາດເຂົ້າໃຈບົດສົນທະນາ ແລະ ສາມາດລົມກັນທາງໂທລະສັບໄດ້ຢ່າງປົກກະຕິ

2. BEAM

- ສ້າງໂດຍ ມາກ ທິນເດນ (Mark W. Tilden) ນັກວິທະຍາສາດ ປະຈຳຫ້ອງທົດລອງແຫ່ງຊາດ LosAlamos ລັດ ນິວແມັກຊິໂກ,ສະຫະລັດອາເມລິກາ
- ສ້າງມາຈາກວົງຈອນອີເລັກໂທນິກຂະໜາດນ້ອຍ, ໃຊ້ອຸປະກອນໜ້ອຍອັນຈຶ່ງມີຂະໜາດນ້ອຍແລະຮູບແບບການເຮັດວຽກບໍ່ຊັບຊ້ອນ. ມີການເຄື່ອນໄຫວຄ້າຍຄືພຶດຕິກຳຂອງສິ່ງມີຊີວິດ ເຊັ່ນ: ມົດ ແລະ ແມງໄມ້ຕ່າງໆ
- " ບີມ " ໃຊ້ລະບົບຄວບຄຸມອີເລັກໂທນິກແບບງ່າຍ ໆ ທີ່ເອີ້ນວ່າ " ເຄືອຂ່າຍເສັ້ນປະສາດ (Nervous Network) " ແທນໄມໂຄຣໂປຣເຊດເຊີ ເຊິ່ງເປັນຊຸດທຣານຊິດເຕີຫຼາຍໆໂຕທີ່ສາມາດຮັບ-ສົ່ງຂໍ້ມູນຈາກໂຄງສ້າງຕົວຫຸ່ນ ແລະ ການເຄື່ອນໄຫວ. ຖ້າຂາຂ້າງໃດກະທົບມໍເຕີໄຟຟ້າ; ຈະເກີດແຮງໜ່ວງ ແລະ ປັບປ່ຽນວົງຈອນໂຄງໄຟຟ້າ, ເຮັດໃຫ້ຂາຂ້າງນັ້ນກ້າວໄປທາງອື່ນທັນທີ

3. ການຈື່ຈຳຕົວເລກ 0-9 ໂດຍໃຊ້ ນິວຣອນເນັດເວີກ

- ເປັນບົດນິພົນທີ່ສະເໜີການໃຊ້ຄວາມຄິດໃນການອອກແບບ ແລະ ສ້າງລະບົບຄອມພິວເຕີໃຫ້ມີໂຄງສ້າງທາງສະຖາປັດຕະຍະກຳຮຽນແບບການເຮັດວຽກຂອງເຊວໃນສະໝອງມະນຸດ (Nerve cell) ຫຼື ນິວຣອນ (Neural)
- ຂຽນໂປຣແກຣມເພື່ອຮັບ input pattern ຂອງຕົວເລກໃນຮູບບິດແມບ ເປັນພາບຂາວດຳຂະໜາດ pattern 16 ຈຸດ16 ພິກເຊລ ໃຊ້ຕົວເລກຕົວພິມໃຫຍ່ ຕົວພິມນ້ອຍ ຕົວພິມອຽງຊ້າຍ ແລະຕົວພິມອຽງຂວາທັງໝົດ 33 pattern ໃຊ້ເປັນຖານຂໍ້ມູນໃນການສອນ ຈາກນັ້ນກຳການກຳນົດນ້ຳໜັກ ,ຄ່າໄບແອດ , Layer , ຟັງຊັນຕ່າງໆ ແລະ output ທີ່ເໝາະສົມ ແລ້ວຈະໄດ້ຄ່າອອກມາຄ່າໜຶ່ງ ຈາກນັ້ນນຳ input ທີ່ຕ້ອງການກວດສອບມາ Simulate ປຽບທຽບກັນວ່າມີຄ່າໃກ້ຄຽງກັບຄ່າໃດ ເມື່ອ Simulate ແລ້ວໄດ້ຄ່າໃດອອກ ມາສ້າງການໂຫຼດ pattern ຄຳຕອບນັ້ນອອກມາສະແດງ

ບົດທີ 7 ລະບົບຜູ້ຊ່ຽວຊານ

ລະບົບຜູ້ຊ່ຽວຊານຈະຖືກໃຊ້ເຮັດວຽກທີ່ຊັບຊ້ອນຫຼາຍທີ່ສຸດ, ເຊິ່ງໃນອະດີດວຽກປະເພດນີ້ຈະສາມາດເຮັດໄດ້ກໍ່ຕ້ອງອາໄສຜູ້ຊ່ຽວຊານທີ່ເປັນມະນຸດເທົ່ານັ້ນ, ດ້ວຍວິທີການປະຍຸກໃຊ້ດ້ານປັນຍາປະດິດ. ລະບົບຜູ້ຊ່ຽວຊານຈະຮັບເອົາຄວາມຮູ້ພື້ນຖານເຊິ່ງມະນຸດເປັນຜູ້ໃສ່ໃຫ້ ມາເຮັດການປະມວນຜົນເຊັ່ນດຽວກັບທີ່ມະນຸດແກ້ບັນຫາທີ່ຊັບຊ້ອນ. ສິ່ງທີ່ດີທີ່ສຸດ ແລະ ມີປະສິດທິພາບຫຼາຍທີ່ສຸດຂອງລະບົບຜູ້ຊ່ຽວຊານແມ່ນການບົ່ງມະຕິຄວາມຮູ້ນັ້ນເຮັດໄດ້ດີກວ່າລະບົບຊອບແວຄອມພິວເຕີທຳມະດາ, ທີ່ໂດຍປົກກະຕິແລ້ວຈະອາໄສມະນຸດເປັນຜູ້ຕັດສິນໃຈ.

ລະບົບຜູ້ຊ່ຽວຊານໄດ້ຖືກນຳໃຊ້ຢ່າງກ້ວາງຂວາງໃນການວິໄຈ, ການວາງແຜນ, ການອອກແບບ, ການແປ, ການຄວບຄຸມ, ການບອກສະຖານະ, ການຄາດການ ແລະ ການອອກຄຳສັ່ງໃນອະນາຄົດ. ດ້ວຍສະຖາປັດຕະຍະກຳສະໄໝໃໝ່ຂອງຮາດແວ (Hardware) ທີ່ຖືກພັດທະນາໃຫ້ໃຊ້ໄດ້ໂດຍກົງກັບລະບົບຜູ້ຊ່ຽວຊານ ແລະ ເຕັກໂນໂລຍີຂອງປັນຍາປະດິດລວມເຂົ້າກັນ. ຄວາມເປັນໄປໄດ້ທີ່ເຮັດໃຫ້ການພັດທະນາລະບົບສາມາດເຮັດວຽກໄດ້ຄືກັບມະນຸດຈຶ່ງມີຫຼາຍຍິ່ງຂຶ້ນ.

ການພັດທະນາລະບົບດັ່ງກ່າວຈະສາມາດເຮັດໃຫ້ເຮົາບໍ່ພຽງແຕ່ມີລະບົບທີ່ມີຄວາມສາມາດຂຶ້ນເທົ່ານັ້ນ ແຕ່ຍັງຈະເຮັດໃຫ້ເຄື່ອງສາມາດເຂົ້າໃຈເລື່ອງທີ່ເກີດຂຶ້ນໄດ້ພ້ອມ.

7.1. ປະຫວັດຂອງລະບົບຜູ້ຊ່ຽວຊານ

ປະຫວັດຂອງລະບົບຜູ້ຊ່ຽວຊານແມ່ນເລີ່ມໃນຊ່ວງປີ 1969 ເອັດເວີດ ໄຟເຈນບາມ (Edward Feigenbaum) ໄດ້ຮ່ວມກັບ ໂຈຊົວ ລີເດີເບີກ (Joshua Lederberg) ຜູ້ທີ່ໄດ້ຮັບລາງວັນໂນເບວ (Nobel) ສາຂາຊີວະເຄມີ ແລະ ບຣູດ ບູຊານັນ (Bruce Buchanan) ໃນການພັດທະນາຊອບແວທີ່ໃຊ້ສຳລັບການວິເຄາະໂຄງສ້າງໂມເລກຸນຂອງສານປະກອບທາງເຄມີໃນປີ ພສ 2508 ທີ່ມະຫາວິທະຍາໄລສະແຕນຟອດ ຊື່ DENDRAL ແລະ ລະບົບຊອບແວນີ້ການເປັນລະບົບຜູ້ຊ່ຽວຊານລະບົບທຳອິດຂອງໂລກ ທີ່ສາມາດເຮັດການວິເຄາະເພື່ອຄາດການໂຄງສ້າງໂມເລກຸນຂອງສານປະກອບ ໂດຍໃຊ້ຄຳສັ່ງ if-then ພື້ນຖານຈຳນວນໜຶ່ງທີ່ບອກເຖິງຄວາມແຕກຕ່າງຂອງອາຕອມໃນສານປະກອບ (Lindsay, R.K. et al., 1993)

ກາງປີ 1970 ເອັດເວີດ ຊໍຕລິຟ (Edward Shortliffe) ນັກຟີຊິກ ແລະ ຄອມພິວເຕີໄດ້ພັດທະນາລະບົບຊອບແວຊະນິດໃໝ່ຊື່ MYCIN ທີ່ວິທະຍາໄລການແພດຂອງມະຫາວິທະຍາໄລສະແຕນຟອດ (Shortliffe, E.H., 1976) ເພື່ອໃຊ້ສຳລັບການບົ່ງມະຕິການຕິດເຊື້ອທີ່ສະໝອງຕໍ່ມາລະບົບນີ້ໄດ້ພັດທະນາກາຍເປັນເປືອກລະບົບຜູ້ຊ່ຽວຊານ (Expert System Shell) ແລະ ພັດທະນາເປັນລະບົບຜູ້ຊ່ຽວຊານລະບົບໃໝ່ ທີ່ໃຊ້ເຮັດການບົ່ງມະຕິຄວາມຜິດປົກກະຕິຂອງປອດຊື່ວ່າ PUFF (Aikins, J.S., et al., 1982)

ໃນປີ ພ.ສ. 2524 ເອັດເວີດ ໄຟເຈນບາມ ແລະ ໂລເບີດ ເອນເຈລມໍ (Robert Engelmore) ແລະ ໝູ່ໄດ້ຮ່ວມກັນຕັ້ງ ບໍລິສັດເຕັກໂນເລດ (Teknowledge) ຂຶ້ນ ຊຶ່ງເປັນບໍລິສັດທຳອິດທີ່ເຮັດລະບົບຜູ້ຊ່ຽວຊານເພື່ອການຄ້າ

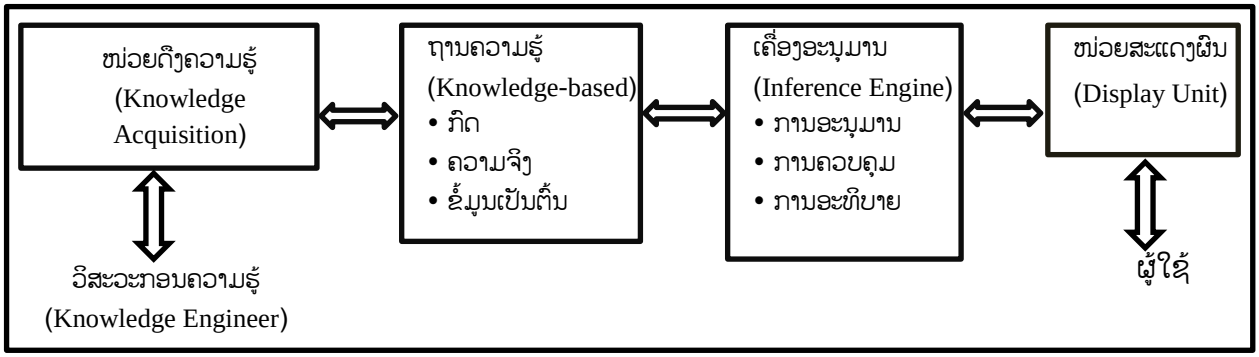
7.2. ນິຍາມຂອງລະບົບຜູ້ຊ່ຽວຊານ

ລະບົບຜູ້ຊ່ຽວຊານແມ່ນການເຮັດໃຫ້ຄອມພິວເຕີມີຄວາມສາມາດໃນການແກ້ປັນຫາທີ່ຊັບຊ້ອນ ໄດ້ ເຊັ່ນດຽວກັບມະນຸດທີ່ເປັນຜູ້ຊ່ຽວຊານ ຈະເຮັດແນວນັ້ນໄດ້ລະບົບຄອມພິວເຕີຈະຕ້ອງຈຳລອງຂະບວນການ ຫາເຫດຜົນຂອງມະນຸດໂດຍອາໄສຄວາມຮູ້ ແລະ ການວິເຄາະ. ຕົວຢ່າງ: ການໃຊ້ລະບົບຜູ້ຊ່ຽວຊານ ສາມາດປຽບທຽບໄດ້ດີກັບການໄປພົບທ່ານໝໍ, ການໄປພົບທ່ານໝໍຕອນເຮົາບໍ່ສະບາຍທ່ານໝໍຈະຕັ້ງຄຳ ຖາມແລ້ວໃຫ້ຄົນເຈັບຕອບ ແລະ ອາດຈະມີການກວດຮ່າງກາຍພ້ອມ ຈາກນັ້ນທ່ານໝໍກໍ່ຈະບົ່ງມະຕິວ່າຄົນ ເຈັບເປັນພະຍາດຫຍັງ. ທ່ານໝໍເຮັດແບບນັ້ນເພາະວ່າເພິ່ນມີຄວາມຮູ້ກ່ຽວກັບພະຍາດ, ເຊິ່ງການສອບຖາມ ອາການຂອງຄົນເຈັບແມ່ນເພື່ອໃຫ້ເປັນຂໍ້ມູນເພື່ອບົ່ງມະຕິພະຍາດດ້ວຍຄວາມຮູ້ທີ່ຕົນມີ. ໃນກໍລະນີນີ້, ຖ້າ ທ່ານໝໍມີຄວາມຮູ້ຫຼາຍການບົ່ງມະຕິພະຍາດຈະມີຄວາມແນ່ນອນກວ່າທ່ານໝໍມີຄວາມຮູ້ໜ້ອຍ. ໃນລະບົບຜູ້ ຊ່ຽວຊານກໍ່ເຊັ່ນດຽວກັນ, ລະບົບຈະຖາມຄຳຖາມຜູ້ໃຊ້ ແລະ ຜູ້ໃຊ້ຈະຕ້ອງຕອບຄຳຖາມ ເມື່ອໝົດຄຳຖາມ ແລ້ວ ຜົນການວິເຄາະຈະອອກມາເປັນຄຳຕອບລາຍງານໃຫ້ຜູ້ໃຊ້ຮັບຮູ້. ໃນລະບົບຜູ້ຊ່ຽວຊານຖ້າລະບົບມີ ເງື່ອນໄຂ (rule) ແລະ ຄວາມຮູ້ຫຼາຍ ການບົ່ງມະຕິພະຍາດຈະມີຄວາມຖືກຕ້ອງກວ່າລະບົບມີຄວາມຮູ້ໜ້ ອຍ. ລັກສະນະພາຍໃນຂອງລະບົບຜູ້ຊ່ຽວຊານຈະປະກອບດ້ວຍຄວາມສາມາດທີ່ສຳຄັນຕ່າງໆດັ່ງຕໍ່ໄປນີ້:

- ຄວາມຮູ້ສະເພາະທີ່ມີ domain ທີ່ເຮົາສົນໃຈ
- ການປະຫຍຸກໃຊ້ວິທີຄົ້ນຫາຂໍ້ມູນ
- ການໃຊ້ການວິເຄາະທາງຫີວຣິດຕິກ (heuristic) ມາຊ່ວຍສະໜັບສະໜູນ
- ຄວາມສາມາດໃນການປະມວນຜົນເພື່ອຫາຄວາມຮູ້ໃໝ່ຈາກຄວາມຮູ້ເກົ່າທີ່ມີຢູ່ແລ້ວ
- ການປະມວນຜົນສັນຍາລັກ (Symbolic Processing)
- ຄວາມສາມາດໃນການອະທິບາຍວິທີການຫາເຫດຜົນ

7.3 ອົງປະກອບຂອງລະບົບຜູ້ຊ່ຽວຊານ

ຈາກແນວຄວາມຄິດ ແລະ ການພະຍາຍາມທີ່ຈະອອກແບບການຄິດ, ການຈື່, ການປະມວນຜົນ ຂອງສະໝອງມະນຸດ ຈຶ່ງໄດ້ມີການອອກແບບລະບົບຜູ້ຊ່ຽວຊານທີ່ແບ່ງອອກເປັນສ່ວນໆ ດັ່ງສະແດງໃນຮູບ ທີ 7.1. ເຊິ່ງເປັນການສະແດງອົງປະກອບຂອງລະບົບຜູ້ຊ່ຽວຊານ ແລະ ເສັ້ນທີ່ເຊື່ອມຫາກັນດ້ວຍລູກ ສອນສະແດງເຖິງໜ່ວຍທີ່ຕິດຕໍ່ກັນຈາກສ່ວນຕ່າງໆ.



ຮູບທີ 7.1 ແຜນວາດຂອງລະບົບຜູ້ຊ່ຽວຊານ

7.3.1 ຖານຄວາມຮູ້ (Knowledge-Based)

ຖານຄວາມຮູ້ແມ່ນສ່ວນຂອງຄວາມຮູ້ທີ່ປະກອບດ້ວຍຄວາມຈິງ ແລະ ກົດຕ່າງໆທີ່ຖືກຈັດໄວ້ໃນ ລັກສະນະຂອງ heuristic ແລະ ມີລັກສະນະໃນການແກ້ບັນຫາສະເພາະບັນຫາໃດໜຶ່ງເຊັ່ນ: ຜູ້ຊ່ຽວຊານ ທີ່ກ່ຽວກັບການຮັກສາໂລກຫົວໃຈ, ໃນຖານຄວາມຮູ້ຈະປະກອບດ້ວຍກົດ ແລະ ຄວາມຈິງທີ່ກ່ຽວກັບເລື່ອງ ຂອງການຮັກສາໂລກຫົວໃຈ ເຊິ່ງກົດ ແລະ ຄວາມຈິງເຫຼົ່ານີ້ຈະຖືກຈັດວາງໄວ້ໃນຖານຄວາມຮູ້ໂດຍ ປົກກະຕິແລ້ວລະບົບຜູ້ຊ່ຽວຊານທີ່ດີຈະສ້າງຖານຄວາມຮູ້ແຍກອອກຈາກລະບົບ, ເພື່ອໃຫ້ຜູ້ສ້າງລະບົບຜູ້ ຊ່ຽວຊານຈະໃສ່ຄວາມຮູ້, ເພີ່ມເຕີມ, ແກ້ໄຂ ຫຼື ປ່ຽນແປງຄວາມຮູ້ອື່ນພາຍຫຼັງ.

ການໃຫ້ຄວາມຮູ້ແກ່ລະບົບຜູ້ຊ່ຽວຊານເອີ້ນວ່າ **ການສະແດງຄວາມຮູ້** (Knowledge Representation), ເນື່ອງຈາກວ່າການສະແດງຄວາມຮູ້ຈະຕ້ອງອາໄສຜູ້ທີ່ມີຄວາມສາມາດໃນການນຳ ຄວາມຮູ້ໃນດ້ານນັ້ນໆ ມາຈັດໃຫ້ຢູ່ໃນຮູບແບບຂອງຄວາມຈິງ ແລະ ກົດຕາມລັກສະນະການປຽບທຽບຂອງ ລະບົບຜູ້ຊ່ຽວຊານ. ການສະແດງຄວາມຮູ້ນີ້ບໍ່ແມ່ນເລື່ອງງ່າຍທີ່ໃຜໆກໍ່ເຮັດໄດ້ ການສະແດງຄວາມຮູ້ເປັນໜ້ າວຽກຫຼັກຂອງນັກວິສະວະກອນຄວາມຮູ້ (knowledge engineering) ຕ້ອງສຶກສາເຖິງວິທີການຫາງ heuristic ຕ່າງໆໃນການແກ້ບັນຫາ ເຊິ່ງຕ່າງຈາກການຂຽນໂປຣແກຣມທຳມະດາ.

7.3.2 ເຄື່ອງອະນຸມານ (Inference engine)

ເຄື່ອງອະນຸມານແມ່ນສ່ວນທີ່ເຮັດໜ້າທີ່ໃນການປຽບທຽບຄວາມຮູ້ຕ່າງໆທີ່ຢູ່ໃນຖານຄວາມຮູ້ ເພື່ອ ເຮັດໜ້າທີ່ໃນການຫາຜົນໄດ້ຮັບທີ່ເປັນໄປໄດ້. ໃນລະບົບຜູ້ຊ່ຽວຊານເຄື່ອງອະນຸມານຈະເຮັດໜ້າທີ່ 2 ຢ່າງ ຄື: 1. ເຮັດໜ້າທີ່ໃນການກວດສອບຄວາມຈິງ ແລະ ກົດທີ່ມີຢູ່ແລ້ວ ແລະ ເພີ່ມຄວາມຈິງອັນໃໝ່ເຂົ້າໄປ ເມື່ອຈຳເປັນ ແລະ 2. ສ້າງການຕັດສິນໃຈກ່ຽວກັບລຳດັບກ່ອນ-ຫຼັງຂອງການອະນຸມານ. ໃນການປະຕິບັດ ສອງໜ້າທີ່ນີ້ເຄື່ອງຈະຕ້ອງເຮັດການຕິດຕໍ່ ແລະ ຂໍຄຳປຶກສາກັບຜູ້ໃຊ້. ອົງປະກອບຂອງເຄື່ອງອະນຸມານ ນັ້ນປະກອບດ້ວຍ 2 ສ່ວນຫຼາຍຄື: ສ່ວນທີ່ກ່ຽວກັບການອະນຸມານ (inference) ໃນການຫາຄວາມຮູ້ໃໝ່

ຈາກຄວາມຈິງ ແລະ ກົດທີ່ມີຢູ່ ແລະ ສ່ວນທີ່ກ່ຽວກັບການຄວບຄຸມ (control) ຈະເຮັດໜ້າທີ່ໃນການຄວບຄຸມ ແລະ ຈັດລຳດັບການອະນຸມານ

❖ **ການອະນຸມານ**

ໃນການອະນຸມານ, ເຄື່ອງອະນຸມານຈະອາໄສຫຼັກການຕ່າງໆດັ່ງຕໍ່ໄປນີ້ ເພື່ອເຮັດການອະນຸມານ:

1. ໂມດັດໂພເນັນ (Modus Ponens) ແມ່ນຍຸດທະສາດໃນການອະນຸມານ ຫຼັກການຂອງໂມດັດໂພເນັນມີວິທີການງ່າຍໆ ຄື: ຖ້າຫາກຮູ້ວ່າ A ຖືກ ແລະ ເມື່ອມີກົດທີ່ວ່າ "If A then B" ເຮົາຈະສາມາດສະຫຼຸບໄດ້ວ່າ B ຖືກ ຫຼື ອີກຄວາມໝາຍໜຶ່ງວ່າ "ເມື່ອພົບວ່າຟຣີມິດ (Premises) ຂອງກົດຖືກຕ້ອງກໍ່ສາມາດເຊື່ອໄດ້ວ່າຂໍ້ສະຫຼຸບ (conclusion) ຂອງກົດຂໍ້ນັ້ນຖືກຕ້ອງດ້ວຍ".

2. ການຫາເຫດຜົນພາຍໃຕ້ຄ່າຄວາມເຊື່ອໝັ້ນ (Certainty factor) ໃນກໍລະນີທີ່ມີຄວາມຮູ້ທີ່ບໍ່ສາມາດຕັດສິນໃຈວ່າຖືກຕ້ອງຮ້ອຍເປີເຊັນ ເຄື່ອງອະນຸມານຈະເຮັດການອະນຸມານຄວາມຮູ້ນີ້ພາຍໃຕ້ຄວາມບໍ່ໝັ້ນ ຄວາມຮູ້ທີ່ບໍ່ໝັ້ນໃຈຈະຖືກກຳນົດໄວ້ດ້ວຍຄ່າຄວາມເຊື່ອໝັ້ນ (Certainty Factor) ທີ່ຂຽນແທນວ່າ cf, ເຊັ່ນ: ສີຂອງທ້ອງຟ້າ = ສີຟ້າ cf 95 ໝາຍຄວາມວ່າເຮົາໝັ້ນໃຈວ່າ ສີຂອງທ້ອງຟ້າເທົ່າກັບສີຟ້າ 95% (ຈາກ 100) ໃນການຫາເຫດຜົນພາຍໃຕ້ຄວາມໝັ້ນໃຈ ເນື່ອງຈາກວ່າໃນການອະນຸມານກົດຂໍ້ຕ່າງໆຈະຕ້ອງມີຄວາມສຳພັນກ່ຽວເນື່ອງກັນ ດັ່ງນັ້ນເມື່ອມີການກຳນົດຄ່າຄວາມໝັ້ນໃຈໃຫ້ກັບ "ຄວາມຈິງ" ແລະ/ຫຼື "ກົດ" ອັນໃດອັນໜຶ່ງຜົນຂອງມັນຈະໄປກ່ຽວຂ້ອງກັບ "ກົດ" ແລະ "ຄວາມຈິງ" ອັນອື່ນໆດ້ວຍ, ເຊັ່ນ:

Fact: ທ້ອງຟ້າມີສີຟ້າ cf 80 (ໝັ້ນໃຈວ່າທ້ອງຟ້າມີສີຟ້າ 80%)

Rule: if ທ້ອງຟ້າມີສີຟ້າ then ອາກາດແຈ່ມໃສ

ຈາກ "ຄວາມຈິງ" ແລະ "ກົດ" ດັ່ງກ່າວເຮົາບໍ່ສາມາດໝັ້ນໃຈໄດ້ 100% ວ່າມື້ນີ້ "ອາກາດແຈ່ມໃສ" ເນື່ອງຈາກວ່າເຮົາໝັ້ນໃຈວ່າມື້ນີ້ "ທ້ອງຟ້າມີສີຟ້າ" ດ້ວຍຄວາມໝັ້ນໃຈແຕ່ 80% ເທົ່ານັ້ນ (ຈາກ Fact) ຂະບວນການຫາເຫດຜົນພາຍໃຕ້ຄວາມບໍ່ແນ່ໃຈ (Shortliffe, E.H. and Buchann, B.G., 1975) ນັ້ນຖືກພັດທະນາຂຶ້ນມາຄັ້ງທຳອິດ ແລະ ໃຊ້ກັບລະບົບຜູ້ຊ່ຽວຊານທີ່ຊື່ວ່າ MYCIN (Shortliffe, E.H., 1976) ມີຢູ່ຫຼາຍວິທີດັ່ງນີ້:

ການໂຮມກັນຂອງຄ່າຄວາມໝັ້ນໃຈ:

$$cf_{total} = cf_1 + \frac{cf_2(100 - cf_1)}{100}$$

ຕົວຢ່າງ:

1. if main-component = fish then best-color = white cf 50.
2. if sauce = tomato then best-color = white cf 70.
3. main-component = fish.
4. sauce = tomato.

ຜົນໄດ້ຮັບຂອງການຄຳນວນຄ່າຄວາມເຊື່ອໝັ້ນເທົ່າກັບ: best-color = white cf 85

ການກຳນົດສັດສ່ວນຂອງຄວາມໝັ້ນໃຈ

$$cf_{total} = \frac{cf_1 * cf_2}{100}$$

ດັ່ງຕົວຢ່າງ:

1. main-component = meat cf 40.
2. if main-component = meat then best-color = red cf 50.

ຜົນໄດ້ຮັບຂອງການຄຳນວນຄ່າຄວາມໝັ້ນໃຈເທົ່າກັບ: best-color = red cf 20

- **ການສ້າງອຳນາດຈຳແນກ (Resolution)**

ອຳນາດຈຳແນກເປັນການພິສູດວ່າຄວາມຈິງທີ່ເກີດຂຶ້ນໃໝ່ນັ້ນເປັນຈິງ ຫຼື ບໍ່ ຈາກກຸ່ມຂອງຕັກກະສາດທີ່ມີຢູ່ແລ້ວ. ວິທີການຂອງອຳນາດຈຳແນກຈະອາໄສຫຼັກການຂອງຕັກກະສາດ

1. ໃນກໍລະນີທີ່ມີກົດດັ່ງນີ້ if A then B ຈະສາມາດປ່ຽນເປັນຕັກກະສາດໄດ້ເທົ່າກັບ not(A) or B
2. ໃນກໍລະນີທີ່ A ແລະ B ເປັນອົງປະກອບຂອງຕັກກະສາດ ຈະສາມາດສ້າງຕາຕະລາງຄ່າຄວາມຈິງໄດ້ດັ່ງນີ້:

A	B	~(A)	If A then B	~(A) or B
T	T	F	T	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

3. ເມື່ອຮູ້ວ່າ ~(A) or B ແລະ ຮູ້ວ່າ A or C ຈະສາມາດສະຫຼຸບປະໂຫຍກທັງສອງໄດ້ເປັນ B or C ພຽງປະໂຫຍກດຽວໄດ້. ການປະຕິບັດແບບນີ້ເອີ້ນວ່າ ການສ້າງອຳນາດຈຳແນກ ເຊິ່ງມີຂັ້ນຕອນດັ່ງຕົວຢ່າງຕໍ່ໄປນີ້:

ຖ້າມີຄວາມຮູ້ທີ່ຕ້ອງການພິສູດ (ຈາກ Harmon, P. and King, D., 1985) ຄື:

1: if (distance > 5 miles) then (mean=drive)

2: if (mean = drive) then (advice = take a cab)

3: fact: distance > 5 miles.

ການພິສູດ:

ຂັ້ນຕອນທີ 1: ປ່ຽນ if … then … ຂອງຄວາມຮູ້ທີ່ກຳນົດໃຫ້ເປັນຕັກກະສາດດັ່ງຕໍ່ໄປນີ້:

1: not (distance > 5 miles) or (mean=drive)

2: not (mean = drive) or (advice = take a cab)

3: fact: distance > 5 miles.

ຂັ້ນຕອນທີ 2: ໃສ່ສົມມຸດຖານທີ່ເຮົາຕ້ອງການຈະທົດສອບ, ສົມມຸດວ່າເຮົາຕ້ອງການຈະທົດສອບວ່າ ຖ້າເຮົາບໍ່ແນະນຳໃຫ້ “take a cab” ຜົນອອກມາຈະເປັນແນວໃດ.

4: not (advice = take a cab)

ຂັ້ນຕອນທີ 3: ເຮັດການທົດສອບດັ່ງນີ້:

not (distance > 5 miles) or (mean=drive) not (mean = drive) or (advice = take a cab)	ຈາກກົດຂໍ້ທີ 1 ຈາກກົດຂໍ້ທີ 2
not (distance > 5 miles) or (advice = take a cab) (distance > 5 miles)	ຈາກກົດຂໍ້ທີ 3
(advice = take a cab) Not (advice = take a cab)	ຈາກກົດຂໍ້ທີ 4
Null	

ຂັ້ນຕອນທີ 4: ສະແດງວ່າສົມມຸດຖານນີ້ໃຊ້ໄດ້ ເນື່ອງຈາກວ່າສົມມຸດຖານທີ່ຕັ້ງຂຶ້ນມາຂັດແຍ່ງກັບຄວາມເປັນຈິງ. ດັ່ງນັ້ນ, ຜົນທີ່ໄດ້ອອກມາຈະຕ້ອງຄັດແຍ່ງ ໃນກໍລະນີທີ່ສົມມຸດຖານທີ່ຕັ້ງມາຖືກຕ້ອງ ຜົນໄດ້ຮັບອອກມາຈະຕ້ອງບໍ່ຂັດແຍ່ງກັນ.

ຈາກຕົວຢ່າງດັ່ງກ່າວໝາຍຄວາມວ່າຂໍ້ສະຫຼຸບ “take a cab” ນັ້ນຖືກ. ການຫາເຫດຜົນຂອງຄວາມຮູ້ແບບຕັກກະສາດນີ້ໃຊ້ວິທີການຂອງອຳນາດຈຳແນກແທນວິທີການຂອງໂມດັດໂພເນັນ ເຊິ່ງໃຊ້ສຳລັບການສະ ແດງຄວາມຮູ້ແບບກົດ.

❖ ການຄວບຄຸມ

ໜ່ວຍຄວບຄຸມໃນເຄື່ອງອະນຸມານເຮັດໜ້າທີ່ສຳຄັນ 2 ຢ່າງຄື:

1. ຄວບຄຸມການເລີ່ມຕົ້ນອະນຸມານວ່າຈະເລີ່ມຕົ້ນຈາກຈຸດໃດໃນຖານຄວາມຮູ້
2. ຄວບຄຸມການຕັດສິນໃຈວ່າຈະເລືອກກົດເກນຂໍ້ໃດໃນການອະນຸມານຕໍ່ໄປ ເພື່ອຫາຄຳຕອບຫຼືກຳນົດວິທີການເລືອກກົດເກນ ຫຼືຄວາມຈິງ ເພື່ອຈະໄດ້ຄຳຕອບທີ່ຖືກຕ້ອງ.

ໜ້າທີ່ຂອງໜ່ວຍຄວບຄຸມຂອງລະບົບຜູ້ຊ່ຽວຊານຈະເປັນຕົວກຳນົດວິທີການອະນຸມານ, ເຊິ່ງມີ 2 ແບບຄື ການອະນຸມານແບບໄປໜ້າ ແລະ :ກັບຫຼັງ (Forward and Backward chaining) ໂດຍປົກກະຕິແລ້ວລະບົບຜູ້ຊ່ຽວຊານທີ່ຖືກສ້າງຂຶ້ນຈະມີວິທີການແບບໃດແບບໜຶ່ງ ຫຼືປະສົມກັນກໍ່ໄດ້ນອກຈາກນັ້ນ .

ການຄວບຄຸມຍັງເປັນຕົວທີ່ກຳນົດວ່າທິດທາງການຄົ້ນຫາຈະເປັນການຄົ້ນຫາແບບເລິກກ່ອນ(Depth First Search) ຫຼືການຄົ້ນຫາແບບກ້ວາງກ່ອນ (Breadth First Search) ທີ່ໃຊ້ປະກອບການອະນຸມານທີ່ເປັນແບບໄປໜ້າ ຫຼືແບບກັບຫຼັງ.

ຕົວຢ່າງທີ 7.1 ຈາກຖານຄວາມຮູ້ທີ່ເປັນກົດລຸ່ມນີ້ ຖ້າກຳນົດວ່າ food = poultry, source = no ແລະ flavor = dry ໃຫ້ແນະນຳວ່າຈະເລືອກ wine ແບບໃດ ດ້ວຍການອະນຸມານແບບໄປໜ້າ.

Rule-1: if food = meat then color = red

Rule-2: if food = poultry then color = white

Rule-3: if food = fish then color = white

Rule-4: if sauce = yes and taste = sweet then flavor = sweet

Rule-5: if flavor = dry then sweetness = dry

Rule-6: if flavor = medium then sweetness = medium

Rule-7: if flavor = sweet then sweetness = sweet

Rule-8: if color = red and sweetness = dry then wine = cabernet-sauvignon

Rule-9: if color = red and sweetness = medium then wine = gamay

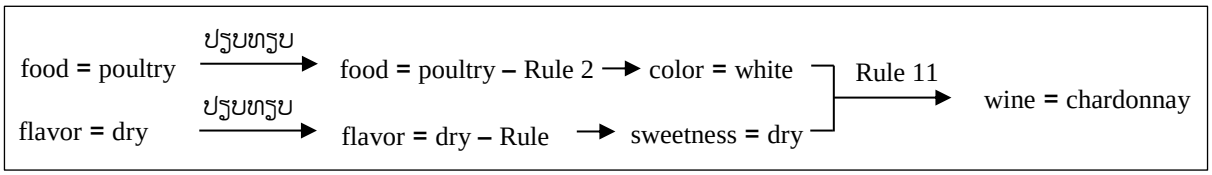
Rule-10: if color = red and sweetness = sweet then wine = burgundy

Rule-11: if color = white and sweetness = dry then wine = chardonnay

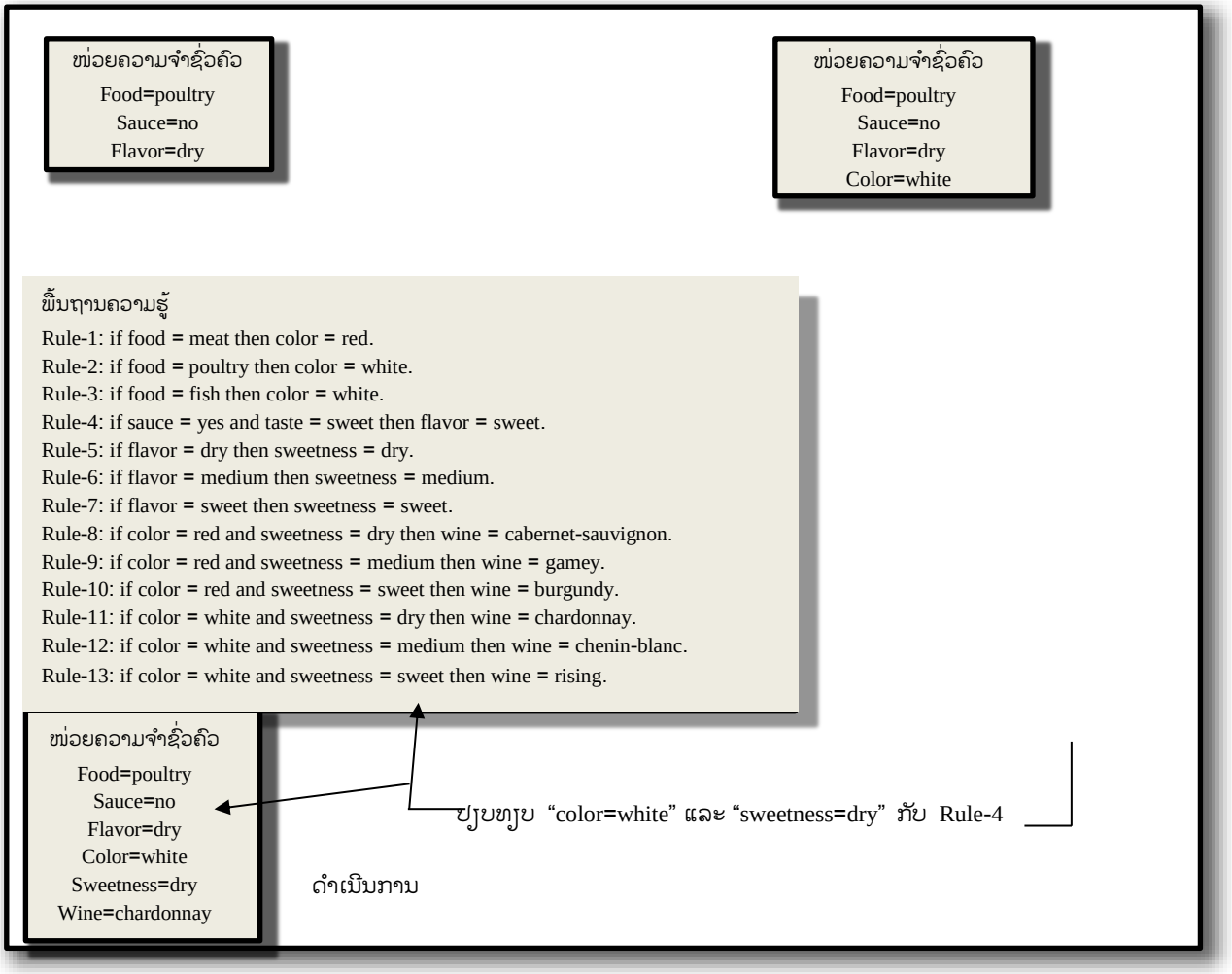
Rule-12: if color = white and sweetness = medium then wine = chenin-blanc

Rule-13: if color = white and sweetness = sweet then wine = Riesling

ການເຮັດວຽກຈະເລີ່ມຈາກສະຖານະປັດຈຸບັນທີ່ເກັບຢູ່ໃນໜ່ວຍຄວາມຈຳ ຄື: food = poultry, sauce = no and flavor = dry ແລ້ວເອີ້ນ food = poultry ໄປປຽບທຽບກັບກົດກ່ອນ ໂດຍເລີ່ມຈາກກົດຂໍ້ທຳອິດ. ການເລີ່ມຈາກກົດຂໍ້ນີ້ເພາະວ່າຢູ່ສ່ວນຫຼັງ if ແມ່ນ food ເມື່ອປຽບທຽບແລ້ວເຮົາຈະເຫັນວ່າຂໍ້ທຳອິດຜິດ ແລະ ທີ່ຖືກແມ່ນຂໍ້ທີ 2 (Rule-2) ເຮົາຈະໄດ້ສ່ວນທີ່ຢູ່ຫຼັງ then ແມ່ນ color = white ເປັນຂໍ້ສະຫຼຸບໃໝ່ ຈາກນັ້ນເຮັດການກວດສອບ sauce = no ເຮັດໃຫ້ກົດຂໍ້ທີ 4 (Rule-2) ຜິດ ແລະ ບໍ່ມີກົດຂໍ້ອື່ນໃຫ້ກວດສອບອີກ. ດັ່ງນັ້ນໄປກວດສອບ flavor = dry ຈະໄດ້ກົດຂໍ້ທີ 5 (Rule-5) ຖືກ ເຮົາຈະໄດ້ຄວາມຮູ້ໃໝ່ທີ່ຢູ່ຫຼັງ then ຄື sweetness = dry ໄປໃຊ້ຕໍ່. ຂໍ້ສະຫຼຸບໃໝ່ທີ່ໄດ້ຄື color = white ແລະ sweetness = dry ແມ່ນເປົ້າໝາຍຍ່ອຍທີ່ເຮົາຈະເອີ້ໄປປຽບທຽບຕໍ່. ເລີ່ມຈາກ color = white ເຫັນວ່າ ກົດທີ່ມີຄຳວ່າ ແມ່ນກົດທີ 8 (Rule-8) ເຖິງ ກົດທີ 13 (Rule-13) ເຊິ່ງ ກົດທີ 11 (Rule-11) ແມ່ນຂໍ້ທຳອິດທີ່ຖືກເພາະມີ color = white ຕໍ່ໄປປຽບທຽບ sweetness ຂອງກົດທີ 11 ຈະໄດ້ sweetness = dry ຕາມຂໍ້ສະຫຼຸບ. ດັ່ງນັ້ນກົດຂໍ້ທີ 11 ຈຶ່ງຖືກ ເຮົາຈະໄດ້ wine = chardonnay ຈົບການເຮັດວຽກຂອງລະບົບ. ຜົນການອະນຸມານໄດ້ສະແດງທີ່ຮູບທີ 7.2 ແລະ ຜັງງານຂອງການດຳເນີນການສະແດງທີ່ຮູບທີ 7.3.



ຮູບທີ 7.2 ສະຫຼຸບຜົນຂອງການອະນຸມານແບບ



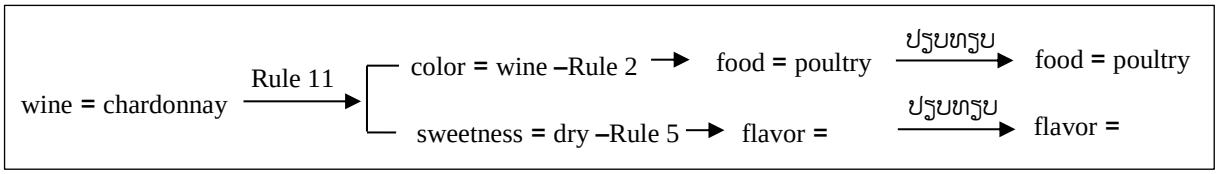
ຮູບທີ 7.3. ຜັງວຽກຂອງການອະນຸມານແບບໄປທາງໜ້າ

ຕົວຢ່າງທີ 7. 2 ຈາກຖານຄວາມຮູ້ທີ່ເປັນກົດເບື້ອງຕົ້ນ ຖ້າກຳນົດວ່າ food = poultry, sauce = no and flavor = dry ໃຫ້ແນະນຳວ່າຈະເລືອກ wine ແບບໃດ ໂດຍໃຊ້ວິທີແບບກັບຫຼັງ.

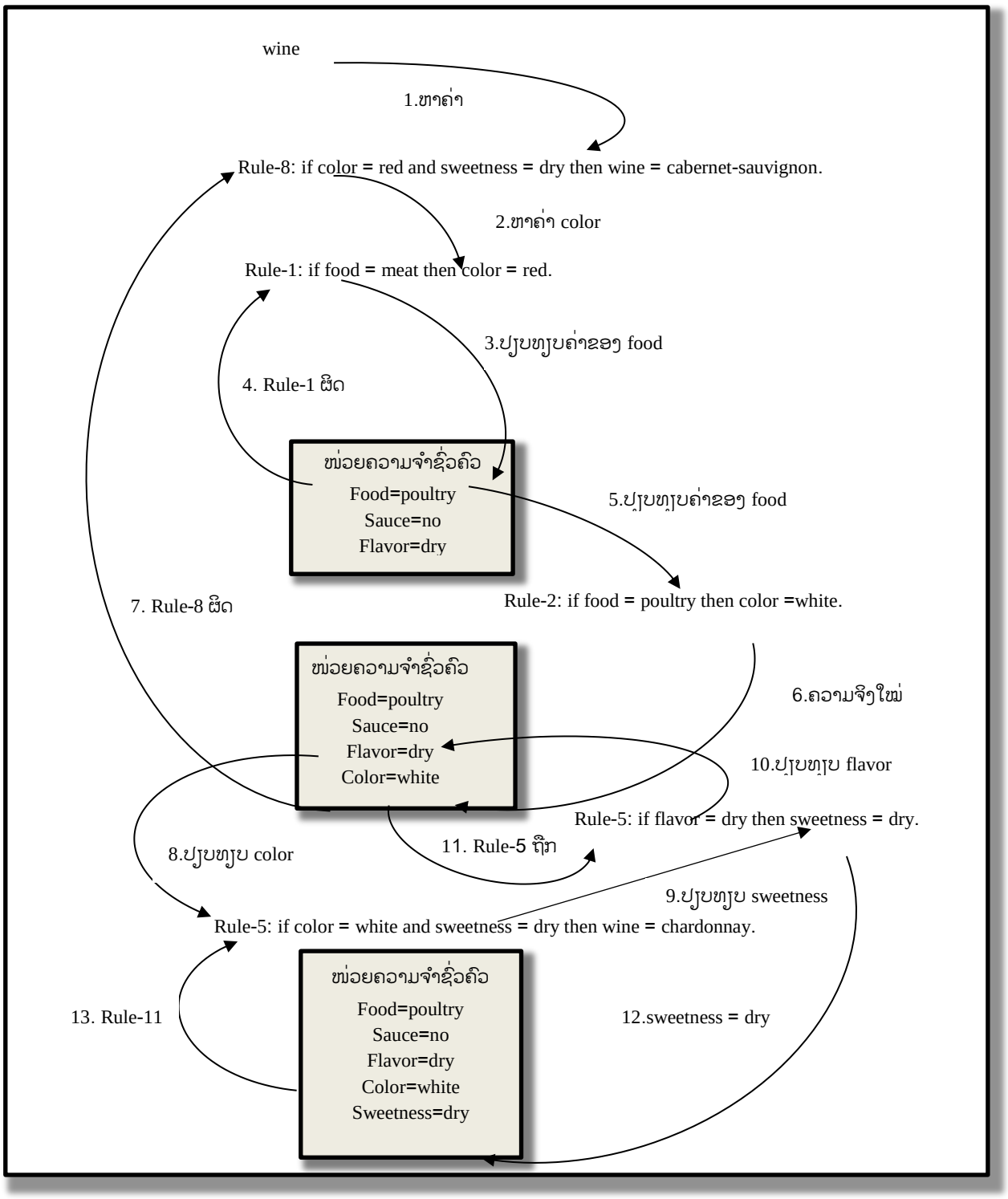
ການອະນຸມານຈະເລີ່ມຈາກການຄົ້ນຫາຄ່າຂອງ wine ເຊິ່ງມາຈາກ goal = wine ໃນຖານຄວາມຮູ້ ດ້ວຍການຫາວ່າມີກົດຂໍ້ໃດແດ່ທີ່ຫຼັງ then ມີການສະຫຼຸບກ່ຽວກັບເລື່ອງ wine. ເມື່ອສຳຫຼວດກໍ່ເຫັນວ່າ ກົດຂໍ້ທີ 8 (Rule-8) ທີ່ຂຽນວ່າ “Rule-8: if color = red and sweetness = dry then wine = cabernet-sauvignon” ຈາກນັ້ນເຮັດການສຳຫຼວດຫຼັງ if ແລ້ວເຮັດການພິສູດວ່າ color = red ເຮັດການພິສູດຫຼັງ then ຕໍ່ມາທີ່ກົດຂໍ້ທີ 1 ດັ່ງນີ້: “Rule-1: if food = meat then color = red” ໃຫ້ຫາຄ່າຂອງ food ເຊິ່ງໄດ້ food = poultry ຈາກຄວາມຈິງທີ່ກຳນົດໃຫ້ໃນຕອນທຳອິດ ເຮັດໃຫ້ກົດຂໍ້ທີ 1 ຜິດ ເຮັດໃຫ້ການເອີ້ນກົດຂໍ້ທີ 2 “Rule-2: if food = meat then color = white”.

ເຊິ່ງຈະໄດ້ food = poultry ທີ່ຖືກຕ້ອງ, ດັ່ງນັ້ນກົດຂໍ້ທີ 2 ຖືກຕ້ອງ ແລະ ເຮົາຈະໄດ້ຂໍ້ສະຫຼຸບໃໝ່ທີ່ວ່າ color = white ເຊິ່ງເຮັດໃຫ້ກົດຂໍ້ທີ 8 ທີ່ກຳລັງພິສູດວ່າ color = red ຜິດທັນທີ. ຈາກນັ້ນເອີ້ນກົດຂໍ້ທີ 9 “Rule-9: if color = red and sweetness = medium then wine = gamay” ທີ່ມີ color = red ກໍ່ຜິດອີກ. ເອີ້ນກົດຂໍ້ທີ 10 “Rule-10: if color = red and sweetness = sweet then wine =

burgundy” ກໍ່ຜິດອີກ. ຈາກນັ້ນເອີ້ນກົດຂໍ້ທີ 11 “Rule-10: if color = white and sweetness = dry then wine = chardonnay” ເຊິ່ງມີ color = white ເຮັດໃຫ້ສ່ວນນີ້ຂອງກົດທີ 11 ຖືກຕ້ອງ. ຈາກນັ້ນ ເຮັດການພິສູດ sweetness ຂອງກົດທີ 11 ໂດຍການເອີ້ນໃຊ້ກົດຂໍ້ທີ 4 “Rule-4: if sauce = yes and taste = sweet then sweetness = sweet” ເຊິ່ງກົດຂໍ້ນີ້ໃຫ້ຫາຄ່າຂອງ sauce ຈາກຄ່າຄວາມຈິງທີ່ກຳນົດ ໄວ້ sauce = no ດັ່ງນັ້ນກົດຂໍ້ທີ 4 ກໍ່ຈະເປັນຜິດ. ເຮົາເອີ້ນໃຊ້ກົດທີ 5 ມາພິສູດຄ່າຂອງ sweetness ຕໍ່ “Rule-5: if flavor = dry then sweetness = dry” ເຊິ່ງກົດຂໍ້ນີ້ໃຫ້ຫາຄ່າຂອງ flavor ແລະ ຄ່ານີ້ຖືກ ກຳນົດໃຫ້ມີຄ່າເທົ່າກັບ dry ດັ່ງນັ້ນກົດຂໍ້ທີ 5 ຖືກຕ້ອງ ແລະ ຈະໄດ້ຄ່າ sweetness = dry ເຮັດໃຫ້ກົດ ຂໍ້ທີ 11 ຖືກພ້ອມ. ດັ່ງນັ້ນເຮົາຈຶ່ງໄດ້ຂໍ້ສະຫຼຸບຂອງ wine = chardonnay ເປັນຄຳຕອບ. ສະຫຼຸບຜົນຂອງ ການອະນຸມານແບບກັບຫຼັງໄດ້ສະແດງໃນຮູບທີ 7.4 ແລະ ຜັງງານຂອງການອະນຸມານໄດ້ສະແດງໃນຮູບ ທີ 7.5.



ຮູບທີ 7.4 ສະຫຼຸບຜົນຂອງການອະນຸມານແບບ



ຮູບທີ 7.5.ຜັງວງກຂອງການອະນຸມານແບບກັບຄືນ

ລັກສະນະສະເພາະອີກແບບໜຶ່ງຂອງເຄື່ອງອະນຸມານທີ່ຈະຕ້ອງພິຈາລະນາກໍ່ຄື ເຄື່ອງອະນຸມານເປັນເຄື່ອງແບບໂມໂນໂທນິກ (Monotonic) ຫຼືແບບນັນໂມໂນໂທນິກ (Nonmonotonic). ການຫາຄວາມຈິງແບບໂມໂນໂທນິກຄື ຄວາມຈິງໃດທີ່ຖືກສຳຫຼວດແລ້ວວ່າເປັນຈິງ ຄວາມຈິງນັ້ນຈະຄົງຢູ່ຕະຫຼອດໄປຈົນກວ່າການໃຫ້ຄຳປຶກສາຂອງລະບົບຈະສິ້ນສຸດ. ຕົວຢ່າງຂອງຄວາມຮູ້ທີ່ຕ້ອງອະນຸມານແບບນີ້, ເຊັ່ນ: ຈາກຄວາມຈິງທີ່ວ່າ “ຄວາມໄວແສງຂອງຟິມເປັນ 100” ຄ່າຂອງຄວາມໄວແສງທີ່ເທົ່າກັບ 100 ນີ້ຈະ

ເປັນຄວາມຈິງຕະຫຼອດການໃຫ້ຄຳປຶກສາ. ສຳລັບການຫາເຫດຜົນແບບນັນໂມໂນໂທນິກຄື ຄວາມຈິງໃດທີ່ເຄີຍຖືກກວດສອບແລ້ວວ່າເປັນຈິງ ຄວາມຈິງນັ້ນຍັງສາມາດຈະປ່ຽນແປງຕໍ່ໄປໃນພາຍຫຼັງໄດ້. ຕົວຢ່າງ ຂອງຄວາມຮູ້ທີ່ຕ້ອງອະນຸມານແບບນີ້ຄື: ການສືບສວນສອບສວນຂອງຕຳຫຼວດໃນການສືບສວນຊ່ວງທຳ ອິດມີຄວາມເປັນໄປໄດ້ວ່າຂໍ້ມູນທີ່ມີຢູ່ຍັງບໍ່ຈະແຈ້ງ ຜູ້ຕ້ອງຫາອາດມີໄດ້ຫຼາຍຄົນ ແຕ່ຫຼັງຈາກທີ່ໄດ້ຮັບຂໍ້ ມູນໃໝ່ໆເຂົ້າມາໃນພາຍຫຼັງແລ້ວ ຜູ້ຕ້ອງຫາອາດຈະປ່ຽນແປງໄປກໍ່ໄດ້.

❖ **ໜ່ວຍດຶງຄວາມຮູ້ (Knowledge Acquisition Unit)**

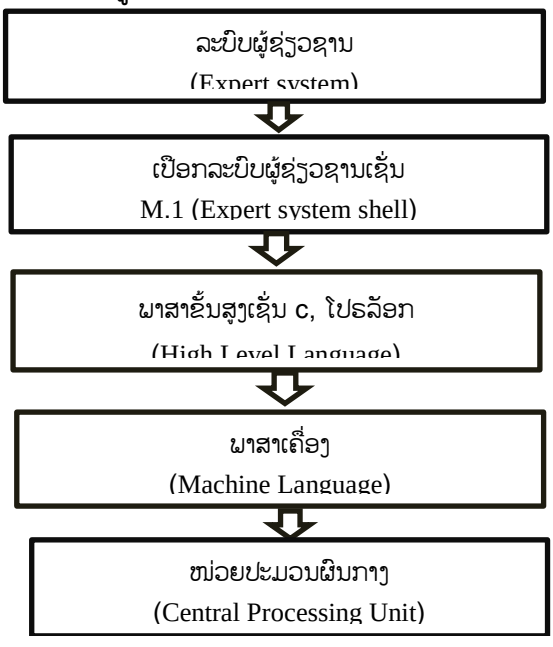
ໜ່ວຍດຶງຄວາມຮູ້ເປັນໜ່ວຍທີ່ຈະຮັບຄວາມຮູ້ຈາກຜູ້ຊ່ຽວຊານ ຫຼື ວິສາວະກອນຄວາມຮູ້ ເມື່ອວິສາ ວະກອນສະແດງຄວາມຮູ້ ໜ່ວຍດຶງຄວາມຮູ້ໃນລະບົບຜູ້ຊ່ຽວຊານຈະເຮັດໜ້າທີ່ແປກົດຄວາມຈິງ, ຂໍ້ສົມມຸດ ຖານ ແລອົງປະກອບອື່ນໆຂອງຄວາມຮູ້ທີ່ມີຢູ່ແຍກຈາກລະບົບເຂົ້າສູ່ຖານຄວາມຮູ້ທີ່ສາມາດປະມວນຜົນໄດ້ ຂອງລະບົບ.

❖ **ໜ່ວຍອະທິບາຍ (Explanation Unit)**

ໜ່ວຍອະທິບາຍແມ່ນໜ່ວຍທີ່ຄອຍອະທິບາຍ ແລະ ໃຫ້ເຫດຜົນໃນການອະນຸມານ ໃນລະຫວ່າງທີ່ຜູ້ ໃຊ້ເຄື່ອງກຳລັງສົນທະນາຢູ່ກັບລະບົບຜູ້ຊ່ຽວຊານນັ້ນ. ຜູ້ໃຊ້ສາມາດຖາມຫາເຫດຜົນໄດ້ວ່າເປັນຫຍັງຈິ່ງຕັ້ງ ຄຳຖາມແບບນັ້ນ ແລະ ຜູ້ໃຊ້ແມ່ນຜູ້ຕ້ອງການຂໍຄຳປຶກສາກັບລະບົບຜູ້ຊ່ຽວຊານ.

7.3.3 ພາສາ ແລະ ເຄື່ອງມື

ໃນລະບົບຜູ້ຊ່ຽວຊານມີແນວຄິດກ່ຽວກັບເລື່ອງພາສາ ແລະ ເຄື່ອງມືທີ່ແຕກຕ່າງຈາກແນວຄິດ ເກົ່າໆຫຼາຍຢ່າງ, ພາສາໃນລະບົບຜູ້ຊ່ຽວຊານຈະໝາຍເຖິງພາສາຂັ້ນສູງ (High Level Language) ທີ່ໃຊ້ ໃນການສ້າງລະບົບຜູ້ຊ່ຽວຊານ, ເຊັ່ນ: ລິສປ (LISP) ແລະ ໂປຣລັອກ (PROLOG) ແລະ ເຄື່ອງມື (Tools) ໝາຍເຖິງເຄື່ອງມືທີ່ໃຊ້ໃນການສະແດງຄວາມຮູ້ໃຫ້ກັບລະບົບຜູ້ຊ່ຽວຊານ, ເຊັ່ນ: EMYCIN ແລະ M.1. ໝາຍຄວາມວ່າພາສາເປັນສ່ວນທີ່ໃຊ້ໃນການສ້າງເຄື່ອງມື ຖ້າຈະສະແດງລະດັບຂອງຊອບແວ ອອກມາເປັນຊັ້ນໆ ຈະໄດ້ດັ່ງຮູບທີ 7.6



ຮູບທີ 7.6 ລະດັບຕ່າງໆຂອງຊອບແວໃນລະບົບຜູ້ຊ່ຽວຊານ

ຈາກຮູບທີ 7.6 ທີ່ສະແດງລະບົບຊອບແວທີ່ມີລະດັບຕ່າງໆ ທີ່ໃຊ້ໃນການສ້າງເປືອກ (shell) ລະບົບຜູ້ຊ່ຽວຊານ M.1 ຈະເຫັນໄດ້ວ່າພາສາຂັ້ນສູງທີ່ໃຊ້ໃນການສ້າງຄືພາສາ C ແລະ ຄວາມຮູ້ກໍ່ໃຊ້ເປືອກລະບົບຜູ້ຊ່ຽວຊານເປັນເຄື່ອງມືເຊັ່ນ: M1. ເປືອກລະບົບຜູ້ຊ່ຽວຊານໃນປັດຈຸບັນມີຫຼາຍຊະນິດ ເຊັ່ນ :Level 5 ເປັນເປືອກລະບົບຜູ້ຊ່ຽວຊານທີ່ໃຊ້ການສະແດງຄວາມຮູ້ໂດຍອາໄສສເຟຣມ (frame) ເປັນຖານ , M.1 ໃຊ້ກົດເປັນຖານ. ດັ່ງນັ້ນ, ໃນລະບົບຜູ້ຊ່ຽວຊານໜຶ່ງໆ ຈະໃຊ້ພາສາຂັ້ນສູງເຊັ່ນ: LISP ແລະ PROLOG, ເປັນໂຕສ້າງເຄື່ອງມືເຊັ່ນ: EMYCIN ແລະ M.1. ຈາກນັ້ນກໍ່ໃຊຊ້ເຄື່ອງມືໃນການສະແດງຄວາມຮູ້. ຜູ້ທີ່ໃຊ້ພາສາຂັ້ນສູງໃນການສ້າງເຄື່ອງມືບໍ່ມີຊື່ເອີ້ນພິເສດ, ແຕ່ສຳລັບຜູ້ທີ່ໃຊ້ເຄື່ອງມືໃນການສະແດງຄວາມຮູ້ມີຊື່ເອີ້ນພິເສດວ່າ “ວິສະວະກອນຄວາມຮູ້”

ການສ້າງລະບົບຜູ້ຊ່ຽວຊານໃນຍຸກທຳອິດບໍ່ໄດ້ມີການແຍກເຄື່ອງມືອອກມາຈາກຕົວລະບົບ (Shortliffe, E.H., 1976) MYCIN ເປັນລະບົບຜູ້ຊ່ຽວຊານລະບົບທຳອິດທີ່ຖືກສ້າງຂຶ້ນມາໂດຍຄະນະຂອງນັກວິຊາການຈາກມະຫາວິທະຍາໄລສະແຕນຟອດ (Stanford) ເປັນໂປຣແກຣມທີ່ໃຫ້ຄຳປຶກສາກ່ຽວກັບການຮັກສາໂລກຫົວໃຈກັບແພດທີ່ຖືກສ້າງຂຶ້ນມາພາຍໃຕ້ພາສາລິສປ ເນື່ອງຈາກ MYCIN ສາມາດເຮັດວຽກໄດ້ສະເພາະກັບການຮັກສາໂລກຫົວໃຈເທົ່ານັ້ນ ເຊິ່ງເປັນຂໍ້ຈຳກັດຫຼາຍ MYCIN ຈຶ່ງຖືກພັດທະນາຕໍ່ມາອີກໂດຍການເອົາຄວາມຮູ້ກ່ຽວກັບໂລກຫົວໃຈອອກ ແລ້ວເຮັດໃຫ້ຜູ້ໃຊ້ໃສ່ຄວາມຮູ້ຫຍັງກໍ່ໄດ້ MYCIN ຈຶ່ງກາຍເປັນຊອບແວໂຕໃໝ່ທີ່ເອີ້ນວ່າ EMYCIN ຫຼື Empty MYCIN ຫຼື MYCIN ວ່າງເປົ່າ. EMYCIN ຈຶ່ງກາຍເປັນເຄື່ອງມືຂອງລະບົບຜູ້ຊ່ຽວຊານຕົວທຳອິດ. ຫຼັງຈາກມີການພັດທະນາ EMYCIN ຕໍ່ມາເຄື່ອງມືທີ່ມີລັກສະນະເຊັ່ນດຽວກັບ EMYCIN ໄດ້ເກີດຂຶ້ນຢ່າງຫຼວງຫຼາຍ ຕົວຢ່າງເຄື່ອງມືໃນປັດຈຸບັນຄື: Level-5 Exsys ແລະ OPS.

ເມື່ອເວົ້າເຖິງພາສາ ແລະ ເຄື່ອງມືກັບລະດັບຕ່າງໆ ຂອງຊອບແວເຫຼົ່ານັ້ນແລ້ວ ບາງຄັ້ງການຈຳແນກລະດັບຂອງສິ່ງຕ່າງໆເຫຼົ່ານີ້ຈະມີຄວາມສັບສົນພໍສົມຄວນມີຊອບແວບາງຊະນິດທີ່ບໍ່ໄດ້ເປັນພາສາຂັ້ນສູງເຊັ່ນດຽວກັບລິສປ ຊຶ່ງໃຊ້ວຽກພຽງແຕ່ດ້ານໂປຣແກຣມເທົ່ານັ້ນ, ແຕ່ກໍ່ບໍ່ໃຊ້ເຄື່ອງມືທີ່ຈະສາມາດສະແດງຄວາມຮູ້ໄດ້ຢ່າງດີເຊັ່ນດຽວກັບ M.1 ຫຼື EMYCIN ເພື່ອຄວາມສະດວກຊອບແວເຫຼົ່ານີ້ຈຶ່ງຖືກຈັດເປັນສິ່ງທີ່ເອີ້ນວ່າ: ສິ່ງແວດລ້ອມ (Environment) ຊອບແວທີ່ມີລັກສະນະເປັນສິ່ງແວດລ້ອມໄດ້ແກ່ OPS5.

ຖ້າຈະພິຈາລະນາກັນໃຫ້ລະອຽດຂຶ້ນໄປອີກ ເມື່ອປຽບທຽບກັບລິສປກັບໂປຣລັອກແລ້ວ ໂປຣລັອກເປັນພາສາຂັ້ນສູງທີ່ສາມາດນຳມາສ້າງລະບົບຜູ້ຊ່ຽວຊານໄດ້ງ່າຍກວ່າລິສປ ຍ້ອນວ່າໂປຣລັອກຈະສາມາດສະແດງຄວາມຮູ້ແບບງ່າຍໆໄດ້ ໃນຂະນະທີ່ລິສປເປັນພຽງພາສາໂປຣແກຣມທຳມະດາ. ດັ່ງນັ້ນ, ໂປຣລັອກຈະມີລັກສະນະໃກ້ສິ່ງແວດລ້ອມຫຼາຍກວ່າລິສປ.

ໃນການເລືອກຊອບແວເພື່ອສ້າງເຄື່ອງມືໃຫ້ກັບລະບົບຜູ້ຊ່ຽວຊານ ຍັງມີການຖຽງກັນຫຼາຍວ່າຈະເລືອກຊອບແວທີ່ເປັນລັກສະນະສິ່ງແວດລ້ອມ ຫຼືໃກ້ສິ່ງແວດລ້ອມ, ເຊັ່ນ: ໂປຣລັອກ, OPS5 ຫຼື ຈະເລືອກພາສາຂັ້ນສູງຢ່າງເຊັ່ນ ລິສປ ຫຼື ຈະເລືອກຊອບແວທີ່ໃກ້ເຄື່ອງເຊັ່ນ: KEE, LOOPS ແຕ່ມີສິ່ງທີ່ນ່າສັງເກດໃນການພິຈາລະນາຢ່າງໜຶ່ງຄື ຊອບແວທີ່ເຂົ້າໃກ້ພາສາຂັ້ນສູງຈະເຮັດໃຫ້ສາມາດສ້າງເຄື່ອງມືທີ່ມີ

ຄວາມຄ່ອງຕົວສູງກວ່າຊອບແວທີ່ເຂົ້າໃກ້ເຄື່ອງມື ແຕ່ຊອບແວທີ່ເຂົ້າໃກ້ເຄື່ອງມືຈະສ້າງໄດ້ງ່າຍກວ່າຊອບແວທີ່ເຂົ້າໃກ້ພາສາຂັ້ນສູງ.

ບົດທີ 8 ການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານ

8.1. ການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານ

ໃນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຈະເປັນການສະແດງຄວາມຮູ້ໂດຍອາໄສເປືອກລະບົບຜູ້ຊ່ຽວຊານ (Expert System Shell) ເປັນວິທີການເກັບຄວາມຮູ້ໃຫ້ກັບລະບົບຜູ້ຊ່ຽວຊານເທົ່ານັ້ນ, ບໍ່ໄດ້ໝາຍເຖິງການສ້າງຊອບແວທີ່ໃຊ້ໃນການເກັບຄວາມຮູ້, ເຊິ່ງເປັນໜ້າທີ່ຂອງນັກຂຽນໂປຣແກຣມທີ່ຈະພັດທະນາຂຶ້ນມາ.

ອີກຢ່າງໜຶ່ງທີ່ຈະຕ້ອງເຂົ້າໃຈກໍ່ຄືລະບົບຜູ້ຊ່ຽວຊານທີ່ເຮົາກ່າວເຖິງແມ່ນລະບົບຜູ້ຊ່ຽວຊານທີ່ສາມາດໃຫ້ຄຳປຶກສາໄດ້ສະເພາະເລື່ອງ, ເຊິ່ງຈະເປັນການຍາກຫຼາຍທີ່ຈະພັດທະນາລະບົບຜູ້ຊ່ຽວຊານທີ່ໃຫ້ຄຳປຶກສາໃນຫຼາຍໆເລື່ອງພາຍໃນຖານຄວາມຮູ້ອັນດຽວກັນ. ໃນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານທີ່ຈະກ່າວເຖິງຕໍ່ໄປນີ້ ຈະກ່າວເຖິງລັກສະນະການພັດທະນາໃນ 2 ລັກສະນະຄື: ການພັດທະນາລະບົບທີ່ມີຂະໜາດນ້ອຍ ແລະ ຂະໜາດໃຫ່ຍ ເຊິ່ງມີລັກສະນະຂອງປັນຫາເປັນການບົ່ງມະຕິຫຼາຍກວ່າການສັງເຄາະ ແລະ ຕົວຢ່າງທີ່ຍົກມາເພື່ອປະກອບການອະທິບາຍຈະເນັ້ນໃສ່ການບົ່ງມະຕິປັນຫາເທົ່ານັ້ນ. ປັນຫາສ່ວນໃຫ່ຍຈະເປັນປັນຫາທີ່ສາມາດປະເມີນຜົນຂອງການປຽບທຽບໄດ້ ແລະ ສາມາດກຳນົດຄຳຕອບທີ່ຄົບຖ້ວນໄດ້. ສຳລັບວິທີການສັງເຄາະເຊິ່ງເປັນປັນຫາທີ່ມີລັກສະນະການຄາດໝາຍຄຳຕອບບໍ່ໄດ້ ຈະບໍ່ໄດ້ກ່າວເຖິງໃນບົດນີ້. ລະບົບຜູ້ຊ່ຽວຊານຂະໜາດນ້ອຍໝາຍເຖິງລະບົບທີ່ມີກົດ (ໃນກໍລະນີທີ່ເປັນ Rule-based) ບໍ່ຫຼາຍ ຖ້າຈະສະແດງຄວາມຮູ້ໃນແບບຮູບຂອງກົດ ຫຼື ຂະໜາດຄວາມຮູ້ທີ່ປະມານກັນເມື່ອສະແດງຄວາມຮູ້ໂດຍວິທີການອື່ນ. ການຕັ້ງກົດເກນວ່າລະບົບຂະໜາດນ້ອຍຄວນມີກົດຈັກຂໍ້ນີ້ຍັງບໍ່ທັນໄດ້ກຳນົດຕາຍຕົວ, ແຕ່ໂດຍຫຼັກສ່ວນໃຫ່ຍແລ້ວຄວນຄຳນຶງເຖິງຂະໜາດຂອງຄວາມຮູ້ ແລະ ຄວາມສັບຊ້ອນຂອງຄວາມຮູ້ພ້ອມ. ກົດເກນງ່າຍໆທີ່ພໍຈະເຂົ້າໃຈໄດ້ລະຫວ່າງລະບົບຜູ້ຊ່ຽວຊານຂະໜາດນ້ອຍ ແລະ ຂະໜາດໃຫ່ຍມີຄວາມແຕກຕ່າງກັນທີ່ເຫັນໄດ້ຈະແຈ້ງຄື ຂອບເຂດຂອງຄວາມຮູ້ມີຄວາມກ້ວາງໃຫ່ຍຂະໜາດໃດ.

ໂດຍຫຼັກການແລ້ວການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານທຸກຂະໜາດມີຫຼັກການດຽວກັນ ອາດຈະມີຄວາມແຕກຕ່າງກັນໃນທາງຂອງວົງຈອນການພັດທະນາ ແລະ ລາຍລະອຽດບາງຢ່າງເທົ່ານັ້ນ. ເນື່ອງຈາກວ່າການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດນ້ອຍ ລະດັບຄວາມສັບຊ້ອນຈະນ້ອຍກວ່າເມື່ອທຽບກັບລະບົບທີ່ມີຂະໜາດໃຫ່ຍ. ດັ່ງນັ້ນ, ຂັ້ນຕອນບາງຢ່າງທີ່ຈະໃຊ້ໃນການພັດທະນາລະບົບຈິ່ງມີຄວາມຈຳເປັນເພາະຈະເປັນການສິ້ນເປືອງ.

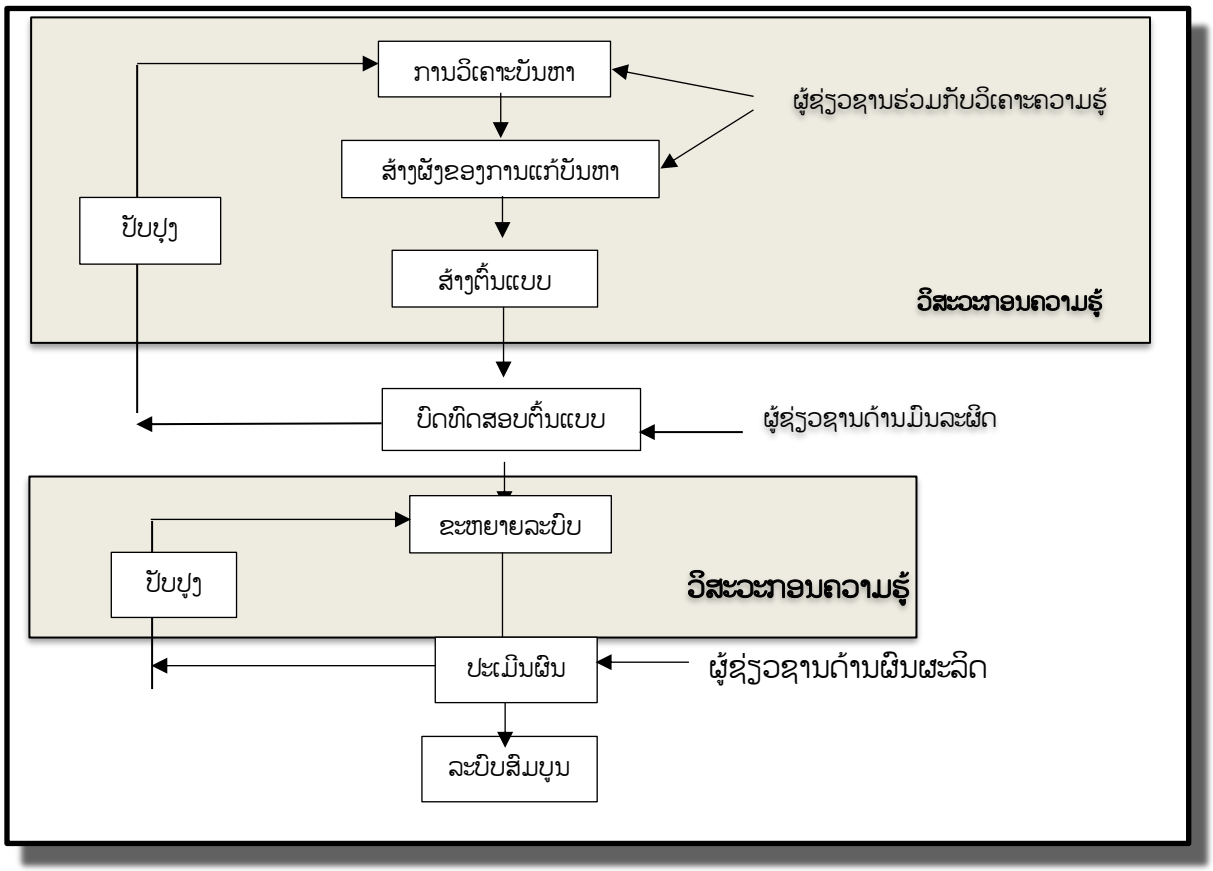
ສຳລັບການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫ່ຍ ວົງຈອນການພັດທະນາຈະຕ່າງກັບລະບົບທີ່ມີຂະໜາດນ້ອຍຫຼາຍ ເນື່ອງຈາກຂະໜາດຂອງລະບົບທີ່ໃຫ່ຍຂຶ້ນ ຄວາມສັບຊ້ອນຂອງລະບົບກໍ່ຕ້ອງມີຫຼາຍຂຶ້ນເຊັ່ນກັນ, ຄວາມຜິດພາດທີ່ອາດເກີດຂຶ້ນ ນັ້ນໝາຍເຖິງເວລາ ແລະ ເງິນລົງທຶນທີ່ເພີ່ມຂຶ້ນຈຳນວນມະຫາສານ. ດັ່ງນັ້ນ, ການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫ່ຍຈິ່ງຕ້ອງມີການວາງແຜນຢ່າງລະມັດລະວັງ ເພື່ອຫຼຸດຜ່ອນຄວາມຜິດພາດທີ່ອາດຈະເກີດຂຶ້ນໃຫ້ໜ້ອຍທີ່ສຸດ.

ໃນຮູບທີ 8.1 ສະແດງວົງຈອນຂອງການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານ ເຊິ່ງຈະເຫັນວ່າໃນການພັດທະນາລະບົບທັງສອງນັ້ນການເຂົ້າຮ່ວມຂອງຜູ້ຊ່ຽວຊານໃນການພັດທະນາລະບົບຈະແຕກຕ່າງກັນ.

ສຳລັບລະບົບຂະໜາດນ້ອຍ ການເຂົ້າຮ່ວມພັດທະນາລະບົບຂອງຜູ້ຊ່ຽວຊານຈະໜ້ອຍກວ່າໃນລະບົບໃຫ່ຍ ຜູ້ທີ່ເຮັດວຽກຫຼັກໆແມ່ນວິສາວະກອນຄວາມຮູ້ (knowledge engineering). ສຳລັບລະບົບຂະໜາດໃຫ່ຍ ຜູ້ຊ່ຽວຊານຈະມີສ່ວນຮ່ວມໃນການພັດທະນາຫຼາຍ, ຜູ້ຊ່ຽວຊານຈະຕ້ອງອຸທິດເວລາຫຼາຍທີ່ສຸດໃຫ້ແກ່ການພັດທະນາລະບົບ ແລະ ອາດຈະຕ້ອງເຂົ້າຮ່ວມໃນການພັດທະນາຢ່າງໃກ້ສິດ.

ການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດນ້ອຍ ຈະເລີ່ມຕົ້ນດ້ວຍການເລືອກເຄື່ອງມື ແລະ ທຳຄວາມເຂົ້າໃຈຕໍ່ກັບປັນຫາກ່ຽວກັບລັກສະນະການໃຫ້ຄຳປຶກສາ. ຈາກນັ້ນກໍ່ເຮັດການຈຳແນກປັນຫາ ແລະ ວິເຄາະຄວາມຮູ້ທີ່ຈະໃສ່ລົງໄປໃນຖານຄວາມຮູ້, ແລ້ວອອກແບບ ແລະ ສ້າງລະບົບຕົ້ນແບບ ໂດຍການກຳນົດຂອບເຂດຄວາມຮູ້ໃຫ້ແຄບລົງ. ແລ້ວຂະຫຍາຍ, ທົດສອບ ແລະ ປັບປຸງລະບົບຈົນກວ່າຈະໃຊ້ໄດ້. ຖ້າຫາກວ່າມີປັນຫາເກີດຂຶ້ນ ເຮົາກໍ່ຈະຍ້ອນກັບໄປທີ່ການຈຳແນກປັນຫາ ແລະ ວິເຄາະຄວາມຮູ້ໃໝ່ ຈົນກວ່າລະບົບຈະເປັນໄປຕາມທີ່ເຮົາຕ້ອງການ.

ຈາກແຜນຜັງຂອງວົງຈອນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫ່ຍ ຈະເຫັນວ່າຜູ້ທີ່ມີບົດບາດໃນການພັດທະນາລະບົບແມ່ນວິສາວະກອນຄວາມຮູ້ ແລະ ຜູ້ຊ່ຽວຊານ. ການພັດທະນາລະບົບສ່ວນໃຫ່ຍຈະຕ້ອງອາໄສການປະສານງານຂອງທັງວິສາວະກອນຄວາມຮູ້ ແລະ ຜູ້ຊ່ຽວຊານ. ວົງຈອນການພັດທະນາຈະເລີ່ມຕົ້ນຈາກການວິເຄາະປັນຫາ, ເລືອກເຄື່ອງມື, ຈາກນັ້ນກໍ່ຈະເປັນການສ້າງລະບົບຕົ້ນແບບ ແລ້ວກໍ່ເຮັດການທົດສອບ ຖ້າຫາກວ່າລະບົບຕົ້ນແບບທີ່ໄດ້ຍັງບໍ່ຖືກຕ້ອງ, ກໍ່ຕ້ອງກັບໄປປັບປຸງລະບົບໃໝ່ ໂດຍເລີ່ມຈາກການວິເຄາະປັນຫາ, ປັບປຸງລະບົບຕົ້ນແບບ ແລະ ທົດສອບໃໝ່ຈົນກວ່າຈະໄດ້ຕົ້ນແບບທີ່ຖືກຕ້ອງ.



ຮູບທີ 8.1 ວົງຈອນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫ່ຍ

ເມື່ອໄດ້ຕົ້ນແບບທີ່ຖືກຕ້ອງແລ້ວ, ເຮັດການຂະຫຍາຍລະບົບໃຫ້ເປັນລະບົບທີ່ສົມບູນ ແລະ ປະເມີນຜົນ. ຖ້າລະບົບທີ່ຂະຫຍາຍຂຶ້ນມາມີຫຍັງຕ້ອງແກ້ໄຂກໍ່ຈະຕ້ອງກັບໄປສ້າງລະບົບທີ່ສົມບູນໃໝ່, ແລ້ວເຮັດການປະເມີນຜົນໃໝ່ຈົນກວ່າຈະໄດ້ຜົນທີ່ພໍໃຈ ເມື່ອໄດ້ລະບົບທີ່ພໍໃຈແລ້ວ ກໍ່ຕິດຕັ້ງລະບົບ ແລະ ວາງແຜນການບຳລຸງຮັກສາເພື່ອເຮັດໃຫ້ລະບົບຜູ້ຊ່ຽວຊານນີ້ມີຄວາມຮູ້ທີ່ທັນສະໄໝ. ລາຍລະອຽດຂັ້ນຕອນຕ່າງໆຂອງການພັດທະນາລະບົບຈະໄດ້ອະທິບາຍດັ່ງຕໍ່ໄປນີ້:

8.1.1. ການຈຳແນກປັນຫາ ແລະ ວິເຄາະຄວາມຮູ້ທີ່ຈະສະໜຸບໃສ່ຖານຄວາມຮູ້

ຄວາມແຕກຕ່າງຂອງການພັດທະນາລະບົບທີ່ມີຂະໜາດນ້ອຍ ແລະ ຂະໜາດໃຫ່ຍອີກຢ່າງໜຶ່ງກໍ່ຄື ໃນລະບົບຂະໜາດນ້ອຍນັ້ນ ວົງຈອນການພັດທະນາຈະເລີ່ມຕົ້ນດ້ວຍການເລືອກເຄື່ອງມື, ແຕ່ສຳລັບການພັດທະນາລະບົບທີ່ມີຂະໜາດໃຫ່ຍ ວົງຈອນການພັດທະນາຈະເລີ່ມຈາກການວິເຄາະປັນຫາ, ເພາະໃນການພັດທະນາລະບົບທີ່ມີຂະໜາດນ້ອຍຈະສາມາດເຮັດໄດ້ໂດຍການເອົາເຄື່ອງມືທີ່ມີຢູ່ມາໃຊ້ ແລະ ນຳຄວາມຮູ້ທີ່ຈະພັດທະນາມາວິເຄາະເພື່ອໃຊ້ກັບເຄື່ອງມືທີ່ມີຢູ່ແລ້ວ ຍ້ອນວ່າຄວາມຮູ້ຂະໜາດນ້ອຍບໍ່ມີຄວາມສັບຊ້ອນຫຼາຍ.

ສຳລັບລະບົບທີ່ມີຂະໜາດໃຫ່ຍ ການພັດທະນາຄວາມຮູ້ຈະຕ້ອງລົງທຶນສູງ ຜູ້ພັດທະນາຈະຕ້ອງພ້ອມທີ່ຈະລົງທຶນຊື້ເຄື່ອງມືໃໝ່ເພາະຄວາມສັບຊ້ອນຂອງລະບົບ. ດັ່ງນັ້ນວົງຈອນການພັດທະນາຈະຕ້ອງເລີ່ມຕົ້ນດ້ວຍການວິເຄາະປັນຫາ, ແລ້ວຈິ່ງເລືອກເຄື່ອງມືທີ່ເໝາະສົມ, ເພາະປັນຫາທີ່ຕ່າງກັນຈະເໝາະກັບເຄື່ອງມືທີ່ຕ່າງກັນ.

ການພັດລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫ່ຍ ຍັງມີເລື່ອງທີ່ຄວນລະວັງອີກຫຼາຍຢ່າງເຊິ່ງຈະໄດ້ກ່າວຕໍ່ໄປຢ່າງລະອຽດໃນຫົວຂໍ້ເລື່ອງການເລືອກປັນຫາທີ່ເໝາະສົມສຳລັບການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫ່ຍ.

ໃນການພັດລະບົບທີ່ມີຂະໜາດນ້ອຍ ເມື່ອໄດ້ເລືອກເຄື່ອງມືແລ້ວຈະຮູ້ວ່າເຄື່ອງມືທີ່ເລືອກໃຊ້ນັ້ນມີລັກສະນະການໃຫ້ຄຳປຶກສາ, ການສະແດງຄວາມຮູ້ ແລະ ຂອບເຂດຂອງປັນຫາເປັນແນວໃດ. ດັ່ງນັ້ນ, ສິ່ງທີ່ຈະຕ້ອງເຮັດຕໍ່ໄປຄືການຈຳແນກປັນຫາ ແລະ ວິເຄາະຄວາມຮູ້, ລັກສະນະການເຮັດວຽກ ແລະ ການວິເຄາະປັນຫາຂອງລະບົບຜູ້ຊ່ຽວຊານມີຫຼາຍວິທີເຊິ່ງຈະໄດ້ກ່າວເຖິງຕາມຫຼັງ.

ການແກ້ໄຂປັນຫາໃນລະບົບຜູ້ຊ່ຽວຊານມີລັກສະນະຄ້າຍຄືກັບການໃຫ້ຄຳປຶກສາທາງໂທລະສັບ ຖ້າສົມມຸດວ່າ ທ້າວ ກ ກຳລັງເຈັບໄສ້ຕິ່ງຢ່າງຮຸນແຮງຕ້ອງໄດ້ຮັບການຜ່າຕັດຢ່າງຮີບດ່ວນ, ຢູ່ເຮືອນທ້າວ ກ ມີທ້າວ ຂ ຢູ່ຄົນດຽວ, ເຊິ່ງບໍ່ມີຄວາມຮູ້ທາງດ້ານການຜ່າຕັດເລີຍ. ທ້າວ ຂ ໂທລະສັບຫາ ທ້າວ ຄ ທີ່ເປັນທ່ານໝໍຊ່ຽວຊານດ້ານການຜ່າຕັດໄສ້ຕິ່ງວ່າຄວນຈະເຮັດແນວໃດ, ຂະນະນີ້ ທ້າວ ຄ ແມ່ນຜູ້ຊ່ຽວຊານທີ່ຈະຕ້ອງຕັ້ງຄຳຖາມ ເພື່ອຖາມ ທ້າວ ຂ ກ່ຽວກັບອາການຂອງ ທ້າວ ກ, ເມື່ອໄດ້ຂໍ້ມູນບາງຢ່າງກໍ່ສາມາດແນະນຳ ທ້າວ ຂ ໃຫ້ປະຕິບັດຕາມຂັ້ນຕອນຂອງການຜ່າຕັດໄສ້ຕິ່ງໄດ້. ການຈຳແນກປັນຫາຂອງ ທ້າວ ຄ ກໍ່ເຊັ່ນດຽວກັບການຈຳແນກປັນຫາເພື່ອເອົາມາໃສ່ໃນຖານຄວາມຮູ້ ຖ້າຫາກວ່າທ່ານຈະຈຳແນກ

ປັນຫາ ທ່ານຕ້ອງມີຄວາມຮູ້ທາງດ້ານນັ້ນຢ່າງດີ ແລະ ສາມາດເຂົ້າໃຈຂະບວນການຕ່າງໆໄດ້ເຊັ່ນດຽວກັບ ທ້າວ ຄ ທີ່ຮູ້ວິທີການຜ່າຕັດໄສ້ຕິ່ງ. ສາມາດສະຫຼຸບໄດ້ດັ່ງນີ້:

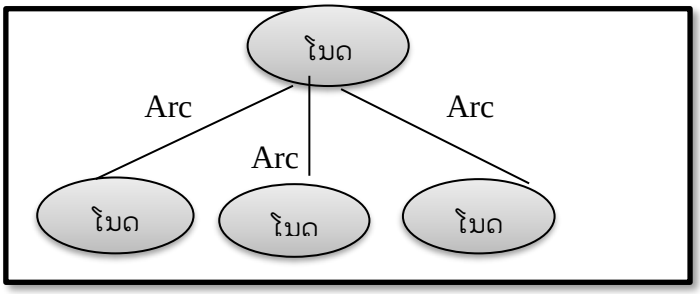
1. **ການມີຄວາມຮູ້ ແລະ ຄວາມເຂົ້າໃຈປັນຫາຢ່າງຖືກຕ້ອງ** ເປັນພື້ນຖານສຳລັບການພັດທະນາລະບົບຄວາມຮູ້, ເຊັ່ນ: ຖ້າຈະສ້າງຖານຄວາມຮູ້ກ່ຽວກັບການຖ່າຍຮູບ ກໍ່ຕ້ອງມີຄວາມຮູ້ກ່ຽວກັບການຖ່າຍຮູບ ແລະ ຈະຕ້ອງເຂົ້າໃຈວ່າໃນຂະບວນການຖ່າຍຮູບນັ້ນມີປັນຫາຫຍັງແດ່. ເງື່ອນໄຂຕ່າງໆ ໃນຄວາມຮູ້ເລື່ອງນັ້ນມີອົງປະກອບຫຍັງທີ່ມີບົດບາດສຳຄັນ, ເຊັ່ນ: ແສງທີ່ຕ່າງກັນ, ການຕັ້ງຄວາມໄວກໍ່ຕ່າງກັນໃນເງື່ອນໄຂຂອງຟິມ (asa) ຕ່າງກັນ ແລະ ການຕັ້ງໜ້າກ້ອງກໍ່ຕ່າງກັນພ້ອມ.

ຕົວຢ່າງທີ 8.1 ຄວາມຮູ້ກ່ຽວກັບການຖ່າຍຮູບ

asa	Light-condition	distance	aperture
100	bright-sun soft-shadow cloudy-bright over-case flash	< 12 > 12	f11 Speed 125 f8 Speed 125 f5.6 Speed 125 f4 Speed 125 f8 Speed 125 f5.6 Speed 125
200	bright-sun soft-shadow cloudy-bright over-case flash	< 12 > 12	F16 Speed 250 f8 Speed 250 f5.6 Speed 250 f4 Speed 250 f11 Speed 250 f8 Speed 250
400	bright-sun soft-shadow cloudy-bright over-case flash	< 12 > 12	F16 Speed 500 F11 Speed 500 f5.6 Speed 500 f5.6 Speed 250 f16 Speed 60 f11 Speed 60

2. **ການຈັດຂັ້ນຕອນເພື່ອແກ້ປັນຫາ** ຜູ້ຊ່ຽວຊານຕ້ອງມີຄວາມເຂົ້າໃຈວ່າຂະບວນການຂອງການແກ້ປັນຫາທັງໝົດເປັນແບບໃດ, ມີວິທີການແນວໃດໃນການແກ້ປັນຫາ, ຂັ້ນຕອນທຸກຂັ້ນຕອນຈະຕ້ອງມີການຈັດລຳດັບ ແລະ ກຳນົດວິທີການແກ້ປັນຫາ. ດັ່ງຕົວຢ່າງທີ 8.1 ໄດ້ສະແດງວິທີການຈັດຂັ້ນຕອນໃນການແກ້ປັນຫາໂດຍອາໄສຕາຕະລາງ. ອີກວິທີການໜຶ່ງທີ່ສາມາດຊ່ວຍໃນການຈັດຂັ້ນຕອນໃນການແກ້ປັນຫາໄດ້ດີຄື ການຈັດຄວາມຮູ້ໃນຮູບແບບຂອງຕົ້ນໄມ້ (tree), ໃນກໍລະນີທີ່ຄວາມຮູ້ມີຂະໜາດໃຫ່ຍ ແລະ ສັບຊ້ອນຫຼາຍ ການໃຊ້ວິທີການຈຳແນກປັນຫາໃນຕົວຢ່າງທີ 8.1 ອາດຈະຫຍຸ້ງຍາກ ການໃຊ້ໂຄງສ້າງຕົ້ນໄມ້ຈະເປັນວິທີທີ່ສະດວກກວ່າ. ອົງປະກອບຂອງໂຄງສ້າງແບບຕົ້ນໄມ້ຈະປະກອບດ້ວຍໂນດ (node) ແລະ ງ່າ (arc), ໂນດຈະແທນຄວາມໝາຍທີ່ສະແດງ

ໃນຖານຄວາມຮູ້ ແລະ ງ່າຈະເປັນສ່ວນເຊື່ອມຄວາມສຳພັນຂອງໂນດຕ່າງໆ ທີ່ກ່ຽວຂ້ອງກັນ ເຂົ້າກັນ.

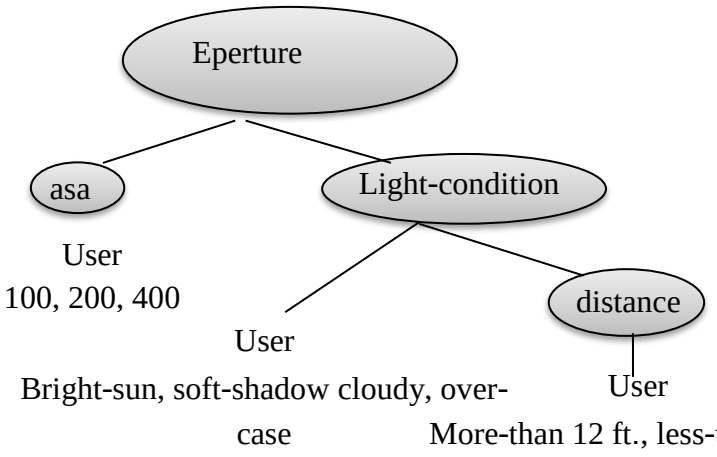


ຮູບທີ 8.2 ໂຄງສ້າງຕົ້ນໄມ້

ໃນສ່ວນຂອງຄວາມສຳພັນລະຫວ່າງໂນດ ຈະກຳນົດໃຫ້ໂນດຕົວທີ່ຢູ່ໃນລະດັບທີ່ສູງກວ່າເປັນ attribute ຂອງໂນດລຸ່ມ ແລະ ໂນດລຸ່ມຈະເປັນຄ່າຂອງໂນດເທິງ ໂດຍມີງ່າເຊື່ອມຄວາມສຳພັນຂອງໂນດທີ່ເປັນ attribute ແລະ ຄ່າເຂົ້າກັນ ເຊິ່ງຈະກຳນົດຄ່າຄວາມສຳພັນໃຫ້ກັບງ່າ ຫຼືບໍ່ກໍ່ໄດ້.

3. **ການຖາມຄຳຖາມເພື່ອເປັນຂໍ້ມູນວ່າຄຳຕອບຈະເປັນແບບໃດ** ປັນຫາສະເພາະໜຶ່ງໆຈະມີວິທີການແກ້ປັນຫາສະເພາະຢ່າງ, ຖ້າປັນຫາເປັນແບບໜຶ່ງການແກ້ປັນຫາຈະເປັນແບບໜຶ່ງ ໃນປັນຫາດຽວກັນຈະມີວິທີການແກ້ປັນຫາໄດ້ຫຼາຍຢ່າງ ແລະ ວິທີການສະເພາະນີ້ຈະຂຶ້ນຢູ່ກັບລັກສະນະສະເພາະຂອງປັນຫາ. ໃນການທີ່ຈະໄດ້ມາເຊິ່ງລັກສະນະສະເພາະຂອງປັນຫາ ເມື່ອໄດ້ລັກສະນະສະເພາະຂອງປັນຫາກໍ່ສາມາດໃຫ້ຄຳປຶກສາໃນການແກ້ປັນຫານັ້ນໄດ້ຢ່າງຖືກຕ້ອງ, ການຕັ້ງຄຳຖາມຕ້ອງງ່າຍຕໍ່ຄວາມເຂົ້າໃຈ ແລະ ເຮັດໃຫ້ຜູ້ໃຊ້ສາມາດຕອບຄຳຖາມໄດ້ຢ່າງຕົງປະເດັນ. ຈາກຕົວຢ່າງທີ 8.2 ໃນສ່ວນຂອງຜູ້ໃຊ້ຄືສ່ວນທີ່ລະບົບຈະຕ້ອງຕິດຕໍ່ກັບຜູ້ໃຊ້ ເຊິ່ງຜູ້ທີ່ສ້າງຖານຄວາມຮູ້ຈະຕ້ອງຕັ້ງເປັນຄຳຖາມທີ່ຈະຕິດຕໍ່ກັບຜູ້ໃຊ້.

ຕົວຢ່າງທີ 8.2 ໂຄງສ້າງແບບຕົ້ນໄມ້ຂອງຖານຄວາມຮູ້ເລື່ອງນັກຖ່າຍຮູບມືໃໝ່



ຮູບທີ 8.3 ໂຄງສ້າງຕົ້ນໄມ້ຂອງຖານຄວາມຮູ້

4. ການໃຫ້ຄຳປຶກສາ ຈະຕ້ອງເປັນລັກສະນະທີ່ເຂົ້າໃຈ ແລະ ສາມາດປະຕິບັດຕາມໄດ້ງ່າຍ ການຕັ້ງຄຳຖາມຈະຕ້ອງຈະແຈ້ງ ໃຫ້ຜູ້ໃຊ້ສາມາດຕອບຄຳຖາມໄດ້ຢ່າງຕົງປະເດັນ.

ຕົວຢ່າງທີ 8.3 ຈາກຕົວຢ່າງທີ 8.1 ແລະ ຕົວຢ່າງທີ 8.2 ຈະເຫັນໄດ້ວ່າຄຳຖາມທີ່ລະບົບຜູ້ຊ່ຽວຊານຕ້ອງການຢາກຮູ້ເພື່ອໃຊ້ໃນການຕັດສິນໃຈຄື:

- ກ່ຽວກັບເງື່ອນໄຂຂອງແສງ (Light-condition)
- ຄວາມໄວແສງຂອງຟິມ (film-speed)
- ໄລຍະຫ່າງ

8.1.2. ການເລືອກເຄື່ອງມື ແລະ ສ້າງຄວາມເຂົ້າໃຈກ່ຽວກັບລັກສະນະຂອງການໃຫ້ຄຳປຶກສາ

ປັດຈຸບັນເຄື່ອງມືທີ່ໃຊ້ໃນການພັດທະນາລະບົບຄວາມຮູ້ມີຫຼາຍຊະນິດ ແລະ ແຕ່ລະຊະນິດກໍ່ມີວິທີການສະແດງຄວາມຮູ້ທີ່ແຕກຕ່າງກັນໄປ. ດັ່ງນັ້ນໃນການເລືອກເຄື່ອງມືແຕ່ລະຊະນິດຈຳເປັນຈະຕ້ອງເລືອກໃຫ້ເໝາະກັບວຽກທີ່ຈະເຮັດ. ນອກຈາກທີ່ກ່າວມາແລ້ວຍັງມີລັກສະນະພິເສດຂອງເຄື່ອງມືແຕ່ລະຊະນິດອີກດ້ວຍ, ເຊັ່ນ: ຄວາມສາມາດໃນການບັນຈຸຄວາມຮູ້, ຄວາມສາມາດໃນການສະແດງພາບ, ສຽງ ແລະ ການຕິດຕໍ່ກັບອຸປະກອນພາຍນອກ ເປັນຕົ້ນ.

ນອກຈາກເຄື່ອງມືທີ່ເຮົາຈະພິຈາລະນາແລ້ວ ໃນສ່ວນຂອງປັນຫາແມ່ນຈຳເປັນທີ່ສຸດທີ່ຈະຕ້ອງພິຈາລະນາເຖິງຂອບເຂດຂອງການໃຫ້ຄຳປຶກສາ (domain) ພ້ອມ. ສຳລັບການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານທີ່ມີຂະໜາດນ້ອຍ ພໍມີວິທີຄ້າວໆທີ່ໃຊ້ເປັນຫຼັກເກນໃນການພິຈາລະນາວ່າລະບົບນີ້ເປັນລະບົບທີ່ມີຂອບເຂດຂະໜາດນ້ອຍໄດ້ດັ່ງຕໍ່ໄປນີ້:

- ເວລາທີ່ໃຊ້ໃນການໃຫ້ຄຳປຶກສາທັງໝົດບໍ່ຄວນເກີນ 30 ນາທີ ນັບຕັ້ງແຕ່ເລີ່ມການຖາມ-ຕອບຄຳຖາມ ຈົນເຖິງຄຳແນະນຳຂັ້ນສຸດທ້າຍ.
- ຄວາມເປັນໄປໄດ້ຂອງຄຳຕອບທີ່ລະບົບຜູ້ຊ່ຽວຊານຈະຕ້ອງເລືອກບໍ່ຄວນເກີນ 50 ຊຸດ.

ຕົວຢ່າງ 8.4: ການພັດທະນາລະບົບຄວາມຮູ້ທີ່ໃຫ້ຄຳປຶກສາກ່ຽວກັບການຖ່າຍຮູບ ໂດຍທົ່ວໄປແລ້ວເຮົາບໍ່ຈຳເປັນທີ່ຈະຕ້ອງອາໄສການສະແດງພາບປະກອບ ແລະ ການສະແດງຄວາມຮູ້ກໍ່ສາມາດໃຊ້ໄດ້ດີກັບ

ການສະແດງໂດຍຮູບແບບຂອງກົດ ເພາະບໍ່ຈຳເປັນຕ້ອງອາໄສການຄຳນວນເຂົ້າມາປະກອບ ຈາກຕົວຢ່າງ ຊ່າງຖ່າຍຮູບມືໃໝ່ເຮົາຈະທົດລອງໃຊ້ M.1 ເຊິ່ງເປັນລະບົບຜູ້ຊ່ຽວຊານຊະນິດ shell ທີ່ເປັນການອະນຸມານແບບກັບຫຼັງ ມີວິທີການສະແດງຄວາມຮູ້ໂດຍອາໄສກົດ ແລະ ມີລັກສະນະການໃຫ້ຄຳປຶກສາແບບຖາມ-ຕອບທີ່ສົມບູນທີ່ສຸດລະບົບໜຶ່ງ. ນອກຈາກນັ້ນແລ້ວ M.1 ຍັງເປັນລະບົບຜູ້ຊ່ຽວຊານຊະນິດ shell ທີ່ສາມາດໃຊ້ໄດ້ກັບການພັດທະນາລະບົບທີ່ມີຂະໜາດນ້ອຍ ແລະ ສາມາດໃຊ້ວຽກກັບເຄື່ອງຄອມພິວເຕີບຸກຄົນ (PC: Personal Computer) ໄດ້.

8.1.3. ການອອກແບບ

ການອອກແບບລະບົບຄວນເລີ່ມຕົ້ນຈາກເຈ້ຍ ຂຽນແນວຄວາມຄິດຂອງຄວາມຮູ້ທັງໝົດທີ່ເຮົາຈະສ້າງ ໂດຍເລີ່ມຕົ້ນຈາກ:

- ຈະຕ້ອງມີເປົ້າໝາຍທີ່ຈະແຈ້ງ, ເປົ້າໝາຍນີ້ໝາຍເຖິງຈຸດໝາຍປາຍທາງຂອງລະບົບວ່າການໃຫ້ຄຳປຶກສານີ້ຈະເປັນແນວໃດ ຫຼືເວົ້າງ່າຍໆກໍ່ຄື ຄຳຕອບຂອງການໃຫ້ຄຳປຶກສານັ້ນເອງ ແລະ ຄຳຕອບນີ້ຈະມີຢູ່ຫຼາຍໆຄຳຕອບ ເຊິ່ງລະບົບຜູ້ຊ່ຽວຊານຈະເປັນຜູ້ເລືອກໃຫ້ສອດຄ່ອງກັບລັກສະນະສະເພາະຂອງປັນຫາ. ຈາກຕົວຢ່າງເລື່ອງ: “ຊ່າງຖ່າຍຮູບມືໃໝ່” ຈະສາມາດກຳນົດໄດ້ວ່າ:

  ຈາກໂຄງສ້າງຕົ້ນໄມ້ໃນຕົວຢ່າງທີ 8.2 ເລື່ອງຊ່າງຖ່າຍຮູບມືໃໝ່ ເປົ້າໝາຍຂອງຄວາມຮູ້ຈະຢູ່ທີ່ໂນດເທິງສຸດ ເຊິ່ງກໍ່ຄື aperture. ດັ່ງນັ້ນ goal = aperture.
- ເງື່ອນໄຂຂອງແສງ
- ເງື່ອນໄຂຂອງ asa
- ໄລຍະຫ່າງ (ຖ້າໃຊ້ເງື່ອນໄຂຂອງແສງ = ໃຊ້ flash)
- ໃຫ້ຄຳປຶກສາ ດັ່ງຕາຕະລາງທີ່ສະແດງຄວາມຮູ້ເບື້ອງຕົ້ນທີ່ໄດ້ກ່າວໄປແລ້ວ

8.1.4 ການສ້າງຕົ້ນແບບ

ໃນການສ້າງຕົ້ນແບບ (Prototype) ເປັນການສະແດງຄວາມຮູ້ສະເພາະຕອນຂຶ້ນມາ ໂດຍການຈຳກັດຂອບເຂດຂອງຄວາມຮູ້ໃຫ້ແຄບລົງ. ໃນການສ້າງລະບົບຕົ້ນແບບນັ້ນ ມີຈຸດປະສົງເພື່ອຫາຄວາມເປັນໄປໄດ້ຂອງການສ້າງລະບົບ ແລະ ຫາຫົນທາງໃນການແກ້ປັນຫາກ່ອນທີ່ຈະສ້າງລະບົບແທ້.

ຕົວຢ່າງທີ 8.5 ຈາກຄວາມຮູ້ກ່ຽວກັບການຖ່າຍພາບໃນຕົວຢ່າງທີ 8.1 ທົດສອບເງື່ອນໄຂ light-condition ເທົ່າກັບ bright-sun ແລະ asa = 100, 200 ຈະໄດ້ຕົ້ນແບບອອກມາດັ່ງນີ້:

goal = advice

1. if light-condition = bright-sun and asa = 100 then
   advice = ‘aperture = f11 and shutter-speed = 125’
2. if light-condition = bright-sun and asa = 200 then
   advice = ‘aperture = f16 and shutter-speed = 250’

question (light-condition) = “What is the condition of light?”

question (asa) = ‘What is the condition of light?’

ລະບົບຕົ້ນແບບທີ່ສ້າງຂຶ້ນນີ້ ຈະຕ້ອງມີລັກສະນະການເຮັດວຽກທີ່ຄືກັບລະບົບຈິງທີ່ຈະພັດທະນາຕໍ່ ແຕ່ກຳໜົດໃຫ້ຂອບເຂດຂອງການແກ້ໄຂບັນຫາເຮັດໄດ້ໜ້ອຍກວ່າ. ລະບົບຕົ້ນແບບນີ້ຈະເປັນຕົ້ນແບບເພື່ອໃຊ້ໃນການທົດສອບວ່າການແກ້ປັນຫາທີ່ໄດ້ເຮັດການອອກແບບມານັ້ນຖືກຕ້ອງ ຫຼືບໍ່ ແລະ ເພື່ອເປັນແນວທາງໃນການຂະຫຍາຍລະບົບຕໍ່ໄປ.

8.1.5 ການຂະຫຍາຍ, ທົດສອບ ແລະ ການປັບປຸງລະບົບ

ການຂະຫຍາຍລະບົບໂດຍການເອົາຕົ້ນແບບທີ່ແນ່ໃຈວ່າຖືກຕ້ອງແລ້ວມາເຮັດການເພີ່ມອົງປະກອບຕ່າງໆ ຈົນເປັນລະບົບທີ່ສົມບູນຕາມທີ່ໄດ້ການວາງແຜນໄວ້ ໂດຍການເພີ່ມຄວາມຮູ້ໃນສ່ວນທີ່ຍັງຂາດຢູ່, ຕ້ອງຕົກແຕ່ງລະບົບໃຫ້ເບິ່ງເປັນແບບປານີດ ແລະ ເພີ່ມສ່ວນທີ່ໃຊ້ໃນການອະທິບາຍສ່ວນຕ່າງໆ.

ໃນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫ່ຍ ກ່ອນທີ່ຈະມີການຂະຫຍາຍລະບົບຕົ້ນແບບນີ້ ຈະຕ້ອງມີການກວດສອບໂດຍຜູ້ຊ່ຽວຊານ ແລະ ວິສາວະກອນຄວາມຮູ້ຢ່າງລະອຽດ ໂດຍການເອົາເງື່ອນໄຂຕ່າງໆທີ່ໄດ້ວາງໄວ້ໃນການສ້າງລະບົບຕົ້ນແບບມາເຮັດການທົດສອບ ແລະ ກວດໂດຍຜູ້ຊ່ຽວຊານ ເພື່ອເບິ່ງວ່າເງື່ອນໄຂທົດສອບນັ້ນຖືກຕ້ອງ ຫຼືບໍ່. ຖ້າຫາກວ່າລະບົບຕົ້ນແບບມີຄວາມຄາດເຄື່ອນຈາກລະບົບທີ່ວາງເອົາໄວ້ກໍ່ຈະຕ້ອງກັບໄປເຮັດການອອກແບບລະບົບຕົ້ນແບບໃໝ່ ສຳຫຼັບການທົດສອບລະບົບຕົ້ນແບບມີສິ່ງທີ່ຕ້ອງຄຳນຶງສະເໝີວ່າລະບົບນີ້ໄດ້ມີການຈຳລອງລະບົບໃຫ້ມີຂອບເຂດຂອງການແກ້ບັນຫາທີ່ໜ້ອຍລົງກວ່າລະບົບຈິງ. ດັ່ງນັ້ນເງື່ອນໄຂໃນການທົດສອບບາງຢ່າງທີ່ບໍ່ໄດ້ກຳນົດໄວ້ໃນການສ້າງລະບົບຕົ້ນແບບກໍ່ຈະນຳມາກວດສອບບໍ່ໄດ້.

ສ່ວນການປະເມີນຜົນຂອງລະບົບເມື່ອລະບົບສ້າງສຳເລັດແລ້ວ ຈະຕ້ອງມີການປະເມີນຜົນວ່າລະບົບທີ່ໄດ້ອອກແບບມານີ້ເປັນໄປຕາມຄວາມຕ້ອງການຂອງຜູ້ອອກແບບລະບົບ ຫຼືບໍ່? ໃນການກວດສອບຜູ້ກວດສອບຈະຕ້ອງມີຜູ້ຊ່ຽວຊານທີ່ມາຊ່ວຍໃນການພັດທະນາ ມາໃຫ້ຄຳປຶກສາຢ່າງໃກ້ຊິດ, ວິສະວະກອນຄວາມຮູ້ຈະຕ້ອງກວດສອບເງື່ອນໄຂຕ່າງໆຂອງການອະນຸມານໃຫ້ຄົບຖ້ວນ ແລະ ຜູ້ຊ່ຽວຊານຈະຕ້ອງກວດສອບຄວາມຮູ້ທຸກຢ່າງທີ່ມີຢູ່ໃນລະບົບວ່າກົງກັບຄວາມເປັນຈິງ ຫຼືບໍ່? ຖ້າຫາກວ່າເກີດຄວາມຜິດພາດ ວິສະວະກອນຄວາມຮູ້ຈະຕ້ອງເປັນຜູ້ແກ້ໄຂກົດ ຫຼືຂໍ້ມູນຕ່າງໆໃນຖານຄວາມຮູ້.

ການບຳລຸງຮັກສາລະບົບຜູ້ຊ່ຽວຊານ, ຜູ້ສ້າງຈະຕ້ອງເຂົ້າໃຈວ່າຄວາມຮູ້ທີ່ໃສ່ເຂົ້າໄປໃນລະບົບຜູ້ຊ່ຽວຊານນັ້ນຫຼ້າສະ ໄໝ ແລະ ປ່ຽນແປງໄດ້ ຈຳເປັນຈະຕ້ອງເພີ່ມເຕີມໃນອະນາຄົດ. ດັ່ງນັ້ນ, ລະບົບຜູ້ຊ່ຽວຊານທີ່ສ້າງຂຶ້ນໃນລະບົບໃຫຍ່ຈຶ່ງຈຳເປັນຫຼາຍທີ່ຈະຕ້ອງມີການເພີ່ມເຕີມຄວາມຮູ້ ຫຼື ຈະຕ້ອງມີການປັບປຸງແກ້ໄຂຢ່າງສະໝ່ຳສະເໝີ ເພື່ອເຮັດໃຫ້ລະບົບຜູ້ຊ່ຽວຊານນີ້ມີຄວາມທັນສະໄໝຢູ່ຕະຫຼອດ.

8.2. ການເລືອກປັນຫາໃຫ້ເໝາະສົມສຳລັບການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫ່ຍ

ມີສິ່ງທີ່ໜ້າສົນໃຈດັ່ງນີ້:

1. ຈຳແນກໂດເມນຂອງປັນຫາ ແລະ ປັນຫາສະເພາະຂອງວຽກ: ໃນການເລືອກປັນຫາທີ່ຖືກຕ້ອງບາງຄັ້ງອາດຈະເປັນສ່ວນທີ່ສຳຄັນທີ່ສຸດຂອງການພັດທະນາລະບົບ. ໃນຂັ້ນຕອນນີ້ເຮົາຈະຕ້ອງເຂົ້າໃຈວ່າເຕັກໂນໂລຢີທາງດ້ານນີ້ຍັງມີຂໍ້ຈຳກັດຫຼາຍ, ຖ້າຫາກການເລືອກປັນຫາບໍ່ຖືກຕ້ອງບາງເທື່ອການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຈະເກີດປັນຫາຂຶ້ນໂດຍທີ່ໃຜກໍ່ບໍ່ສາມາດຊ່ວຍແກ້ໄຂປັນຫາໄດ້. ດັ່ງນັ້ນ, ການເລືອກ

ບັນຫາຈຶ່ງຈຳເປັນຫຼາຍທີ່ຈະຕ້ອງພິຈາລະນາຄວາມສາມາດຂອງລະບົບຜູ້ຊ່ຽວຊານ ເພາະໃນລະບົບໃຫຍ່ຖ້າຫາກວ່າການເລືອກບັນຫາຜິດອາດຈະເຮັດໃຫ້ທັງລະບົບຜິດພາດໄດ້.

2. ຕ້ອງມີຜູ້ຊ່ຽວຊານທີ່ພ້ອມຈະໃຫ້ການຊ່ວຍເຫຼືອ: ດັ່ງທີ່ຮູ້ມາແລ້ວວ່າລະບົບຜູ້ຊ່ຽວຊານເປັນລະບົບທີ່ສ້າງຂຶ້ນມາເພື່ອພະຍາຍາມຮຽນແບບບການເຮັດວຽກຂອງຜູ້ຊ່ຽວຊານ. ດັ່ງນັ້ນ, ໃນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຂະໜາດໃຫຍ່ ຈຶ່ງຈຳເປັນຢ່າງຫຍິ່ງທີ່ຈະຕ້ອງມີຜູ້ຊ່ຽວຊານຊ່ວຍໃຫ້ຄວາມຊ່ວຍເຫຼືອ. ໃນການພັດທະນາລະບົບໃຫຍ່ນີ້ຈະແຕກຕ່າງຈາກລະບົບນ້ອຍ, ເຊິ່ງວິສະວະກອນຄວາມຮູ້ສາມາດປະມວນຄວາມຮູ້ໄດ້ດ້ວຍຕົວເອງ ແລະ ຫາຄວາມຮູ້ເພີ່ມເຕີມຈາກພາຍນອກໄດ້. ແຕ່ໃນລະບົບໃຫຍ່ນັ້ນຄວາມຮູ້ທີ່ໃຊ້ຈະຕ້ອງເປັນຄວາມຮູ້ທີ່ໄດ້ມາຈາກຜູ້ຊ່ຽວຊານທີ່ມີປະສົບການໃນຄວາມຮູ້ນັ້ນຢ່າງດີ, ສາມາດເຂົ້າໃຈ ແລະ ຮູ້ວິທີແກ້ໄຂບັນຫານັ້ນໆໄດ້.
3. ວິເຄາະການແກ້ໄຂປັນຫາໂດຍຫຍໍ້: ການວິເຄາະເປັນສ່ວນທີ່ຈະຊ່ວຍໃຫ້ວິສະວະກອນຄວາມຮູ້ໄດ້ເຂົ້າໃຈແລະ ຫາເຄື່ອງມືທີ່ຖືກຕ້ອງມາໃຊ້.
4. ການແກ້ໄຂປັນນຫາກັບລະບົບທີ່ເກີດຂຶ້ນນັ້ນຈຳເປັນ ຫຼືບໍ່ ທີ່ຈະຕ້ອງອາໄສລະບົບຜູ້ຊ່ຽວຊານມາຊ່ວຍແກ້: ປັນຫາທຸກປັນຫາບໍ່ຈຳເປັນທີ່ຈະຕ້ອງແກ້ໄຂດ້ວຍລະບົບຜູ້ຊ່ຽວຊານສະເໝີໄປ ເຄື່ອງມືທີ່ເປັນໂປຣແກຣມບາງຢ່າງອາດຈະແກ້ໄຂບັນຫາໄດ້ດີກວ່າລະບົບຜູ້ຊ່ຽວຊານກໍ່ໄດ້. ດັ່ງນັ້ນ, ກ່ອນທີ່ຈະເລືອກລະບົບຜູ້ຊ່ຽວຊານມາເປັນຕົວທີ່ຊ່ວຍໃນການແກ້ໄຂບັນຫາ ຄວນຈະກວດສອບເບິ່ງໂປຣແກຣມລະບົບເກົ່າກ່ອນວ່າສາມາດແກ້ໄຂບັນຫາດັ່ງກ່າວໄດ້ຫຼືບໍ່. ສຳຫຼັບລະບົບຜູ້ຊ່ຽວຊານນັ້ນປັນຫາທີ່ເໝາະສົມຈະຕ້ອງເປັນບັນຫາທີ່ກ່ຽວກັບທັກສະ ແລະ ຄວາມຊຳນານ.

8.2.1. ການຈຳແນກຊະນິດຂອງເຄື່ອງມືທີ່ໃຊ້ໃນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານ

ການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານໃນປັດຈຸບັນມີການປ່ຽນແປງຢ່າງວ່ອງໄວທັງໃນແງ່ຂອງຄວາມງ່າຍແລະຄວາມໄວທີ່ໃຊ້ໃນການພັດທະນາລະບົບ ເນື່ອງຈາກການພັດທະນາຢ່າງວ່ອງໄວຂອງເຄື່ອງມືທີ່ໃຊ້ສຳຫຼັບການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານ (Expert System Building Tool : ESBT) ນັ້ນເອງ. ເຄື່ອງມືເຫຼົ່ານີ້ໂດຍສ່ວນຫຼາຍແລ້ວມັກຈະຖືກສ້າງຂຶ້ນມາເພື່ອການຄ້າ ແລະ ມີລັກສະນະພິເສດໃນການນຳໃຊ້ທີ່ຕ່າງກັນຂຶ້ນຢູ່ກັບວິທີການທີ່ໃຊ້ໃນການສ້າງເຄື່ອງມືເຫຼົ່ານີ້. ໃນຫົວຂໍ້ຕໍ່ໄປຂອງບົດນີ້ຈະອະທິບາຍເຖິງລາຍລະອຽດວິທີການຕ່າງໆ ທີ່ໃຊ້ໃນການສ້າງເຄື່ອງມືເພື່ອການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານ, ເຊັ່ນ: ລາຍລະອຽດໂຄງສ້າງຂອງເຄື່ອງມືສະແດງຄວາມຮູ້, ເຄື່ອງອະນຸມານການຕິດຕໍ່ຜູ້ພັດທະນາລະບົບ, ການຕິດຕໍ່ກັບ End-User ແລະ ພາສາທີ່ໃຊ້ໃນການພັດທະນາເຄື່ອງມື.

8.2.2. ໂຄງສ້າງຂອງເຄື່ອງມືທີ່ໃຊ້ໃນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານ

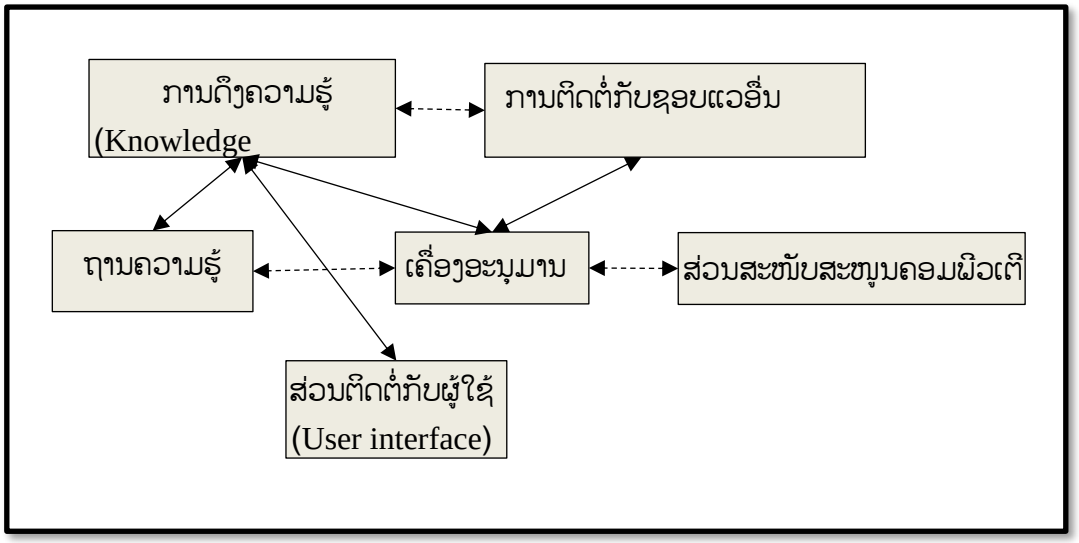
ດັ່ງທີ່ໄດ້ກ່າວໃນບົດທີ່ຜ່ານໆມາວ່າ ໂຄງສ້າງຫຼັກຂອງລະບົບຜູ້ຊ່ຽວຊານຈະປະກອບດ້ວຍຖານຄວາມຮູ້ ແລະ ເຄື່ອງອະນຸມານ ເຊິ່ງເປັນສ່ວນທີ່ເຮັດໜ້າທີ່ໃນການອະນຸມານຄວາມຮູ້ທີ່ມີຢູ່ໃນຖານຄວາມຮູ້ ແລະ ສ່ວນທີ່ໃຊ້ໃນການຕິດຕໍ່ກັບຜູ້ພັດທະນາລະບົບ (Knowledge Acquisition Subsystem) ເຊິ່ງເຮັດ ໜ້າທີ່ໃນການອຳນວຍຄວາມສະດວກຕໍ່ຜູ້ທີ່ຈະມາພັດທະນາລະບົບຜູ້ຊ່ຽວຊານ.

ເມື່ອເບິ່ງສະເພາະສ່ວນຂອງການຕິດຕໍ່ລະຫ່ວາງເຄື່ອງມື (ESBT) ກັບຜູ້ພັດທະນາລະແລ້ວ ຈະເຫັນຄວາມຈຳເປັນຂອງການມີລະບົບນີ້ຂຶ້ນມາຄື:

- ເພື່ອໃຫ້ການພັດທະນາຖານຄວາມຮູ້ທີ່ຕ້ອງການມີຄວາມສອດຄ່ອງກັບລະບົບ
- ເພື່ອໃຫ້ຄວາມສາມາດສ້າງສ່ວນທີ່ເຮັດໜ້າທີ່ຕິດຕໍ່ກັບ End-user ເປັນໄປຕາມທີ່ຜູ້ພັດທະນາລະບົບຕ້ອງການ
- ເພື່ອໃຫ້ສາມາດເພີ່ມສ່ວນທີ່ເຮັດໜ້າທີ່ໃນການຄວບຄຸມການອານຸມານເຂົ້າໄປໄດ້

ຄວາມສາມາດທີ່ກ່າວມາເປັນສິ່ງໜຶ່ງທີ່ບອກລະດັບຂອງເຄື່ອງມືທີ່ໃຊ້ໃນການພັດທະນາລະບົບຜູ້ຊ່ຽວຊານໄດ້ເປັນຢ່າງດີ ວ່າມີຄວາມສາມາດສູງ ຫຼືບໍ່. ສ່ວນນີ້ນອກຈາກໜ້າທີ່ທີ່ໃຊ້ຕິດຕໍ່ກັບຜູ້ພັດທະນາລະບົບແລ້ວໃນສ່ວນຂອງການຕິດຕໍ່ກັບເລື່ອງອື່ນຍັງເປັນສິ່ງທີ່ຈຳເປັນອີກເຊັ່ນກັນຄື:

- ຄວາມສາມາດໃນການຕິດຕໍ່ກັບຊອບແວ ແລະ ຖານຂໍ້ມູນ (Database) ຊະນິດອື່ນ
- ຄວາມສາມາດໃນການໃຊ້ປະໂຫຍດຈາກລະບົບຄອມພິວເຕີເຊັ່ນຄວາມສາມາດໃນການຕິດຕໍ່ກັບເຄືອຂ່າຍຂອງລະບົບສື່ສານເປັນຕົ້ນ



ຮູບທີ 8.4 ໂຄງສ້າງຂອງ ESBT(ເສັ້ນທຶບສະແດງສິ່ງທີ່ສຳພັນກັນໂດຍພື້ນຖານ ແລະເສັ້ນ

❖ ການສະແດງຄວາມຮູ້

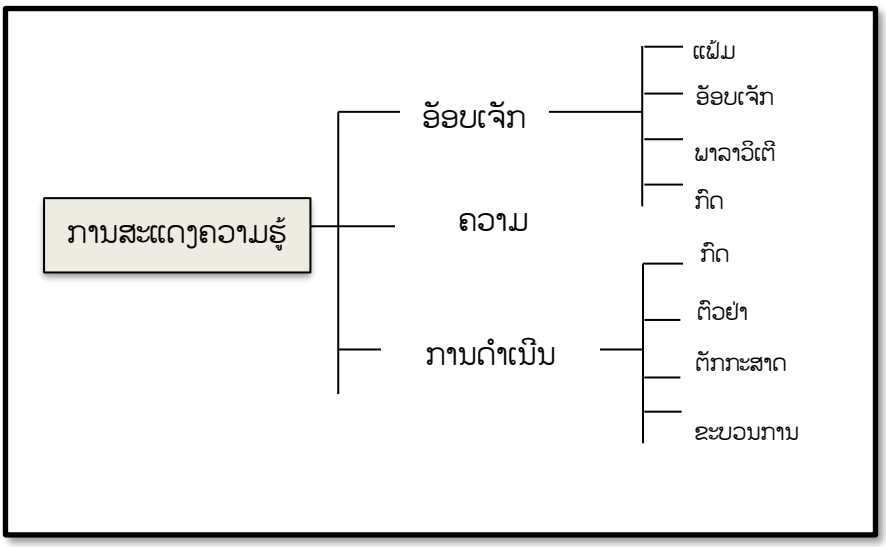
ເງື່ອນໄຂທຳອິດທີ່ໃຊ້ສຳລັບພິຈາລະນາໃນການເລືອກ ESBT ຄືວິທີການໃນການສະແດງຄວາມຮູ້ຈະຕ້ອງງ່າຍ. ຈາກຮູບທີ 8.5 ຈະເຫັນວ່າການສະແດງຄວາມຮູ້ປະກອບດ້ວຍ 3 ຢ່າງຄື: Object Descriptions, Certainties ແລະ Actions. 3 ຢ່າງນີ້ເປັນຫົວຂໍ້ພື້ນຖານຂອງການສະແດງຄວາມຮູ້ ທີ່ຈະຕ້ອງມີເພື່ອປະກອບກັນເປັນຖານຄວາມຮູ້ໃນຖານຄວາມຮູ້ທີ່ເປັນແບບກົດ. ອອບເຈັກຈະຖືກສະແດງໄວ້ໃນສ່ວນຂອງຫຼັງ if ແລະAction ຈະຖືກສະແດງໄວ້ໃນສ່ວນຂອງຫຼັງ then ສຳລັບ Certainties ແມ່ນສ່ວນທີ່ເປັນ cf.

Object Descriptions ເປັນສ່ວນທີ່ກ່ຽວກັບການອະທິບາຍອອບເຈັກທີ່ໃຊ້ໃນຖານຄວາມຮູ້, ວິທີການສະແດງຄວາມຮູ້ທີ່ໃຊ້ທົ່ວໄປແມ່ນເຟຣມ. ໂດຍປົກກະຕິແລ້ວການສະແດງຄວາມຮູ້ແບບນີ້ຈະສາມາດຄວບຄຸມເຖິງເລື່ອງການກຳນົດອອບເຈັກຄູ່ພາລາມິເຕີ, ຕັກກະສາດ ແລະ ກົດໄດ້. ໃນກໍລະນີຂອງ ESBT

ທີ່ມີວິທີການສະແດງຄວາມຮູ້ທີ່ດີແລ້ວໃນຖານຄວາມຮູ້ໜຶ່ງຈະສາມາດມີ Multiple World ໄດ້ໂດຍ ການສະແດງຄວາມຮູ້ທີ່ແບ່ງອອກເປັນເຟຣມຫຼາຍອັນ.

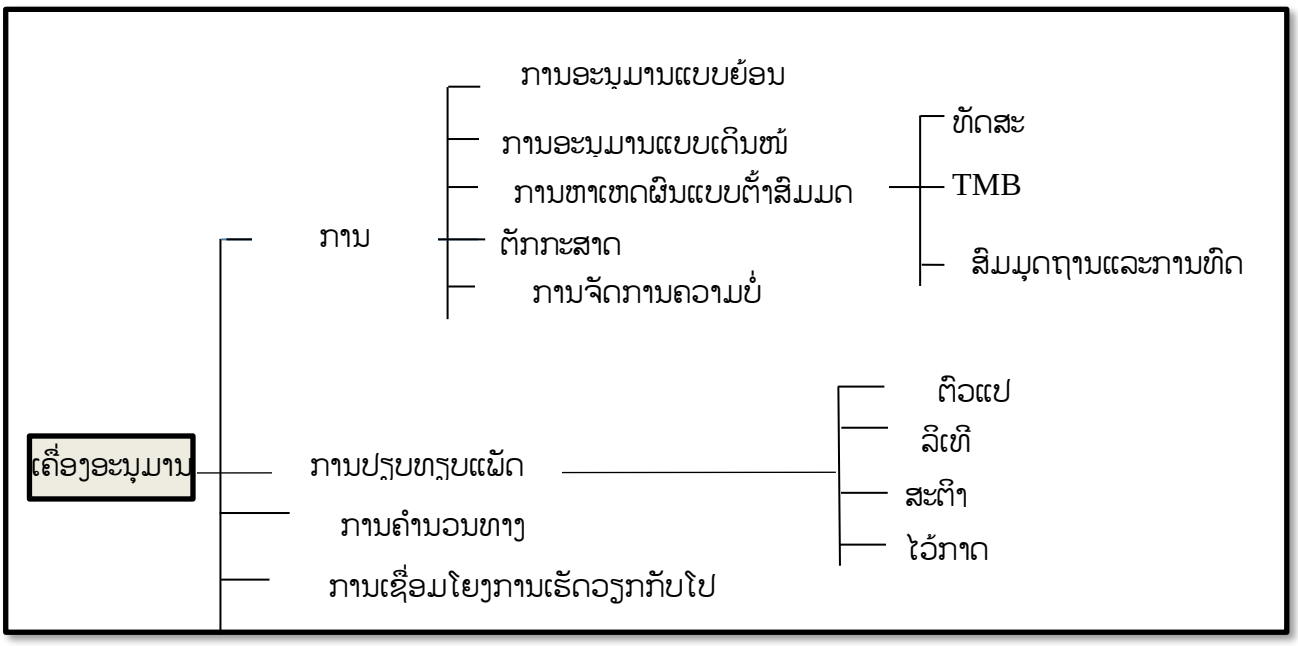
ການດຳເນີນການ (Actions) ເປັນສ່ວນທີ່ປ່ຽນສະຖານະການ ຫຼືຖານຂໍ້ມູນທີ່ກ່ຽວຂ້ອງ. ໃນສ່ວນຂອງ ການດຳເນີນໃນຖານຄວາມຮູ້ນັ້ນສາມາດສະແດງໄດ້ຫຼາຍວິທີ, ແຕ່ໂດຍສ່ວນໃຫ່ຍແລ້ວການດຳເນີນການ ຈະສະແດງຢູ່ໃນຮູບແບບຂອງກົດ ເຊິ່ງກົດເຫຼົ່ານີ້ອາດຈະຖືກນຳມາຈັດເປັນກຸ່ມເພື່ອຄວາມສະດວກໃນການ ປັບປຸງແກ້ໄຂ. ຮູບແບບໜຶ່ງທີ່ນິຍົມໃຊ້ຫຼາຍກໍ່ຄືການໃຊ້ຕົວຢ່າງ, ເຊິ່ງວິທີການສະແດງຄວາມຮູ້ເຊັ່ນນີ້ຈະ ງ່າຍກ່ວ່າແບບກົດຫຼາຍ.

ໃນການສະແດງຄວາມຮູ້, ຜູ້ທີ່ພັດທະນາລະບົບຜູ້ຊ່ຽວຊານຈະຕ້ອງແທນຄ່າຂອງອອບເຈັກ ແລະ ການ ດຳເນີນການປະກອບກັນເປັນຖານຄວາມຮູ້. ການແທນຄ່າອອບເຈັກນັ້ນ, ຜູ້ພັດທະນາລະບົບຈະຕ້ອງ ພິຈາລະນາເຖິງລະດັບຂອງຄວາມໝັ້ນໃຈທີ່ມີຕໍ່ຄວາມຮູ້ນັ້ນດ້ວຍ. ຖ້າຫາກໝັ້ນໃຈໃນຄວາມຮູ້ນັ້ນບໍ່ເຖິງ 100% ລັກສະນະຂອງການສະແດງຄວາມຮູ້ດັ່ງກ່າວຈະເອີ້ນວ່າ: ຄ່າຄວາມໝັ້ນໃຈ (Certainty Factor).



ຮູບທີ 8.5 ວິທີການຕ່າງໆ ຂອງການສະແດງ

❖ **ເຄື່ອງອະນຸມານ**



ຮູບທີ 8.6 ການອະນຸມານຊະນິດຕ່າງໆ

❖ **ການຕິດຕໍ່ກັບຜູ້ໃຊ້**

ສິ່ງທີ່ຈະຕ້ອງພິຈາລະນາໃນສ່ວນຂອງການຕິດຕໍ່ກັບຜູ້ໃຊ້ຄື:

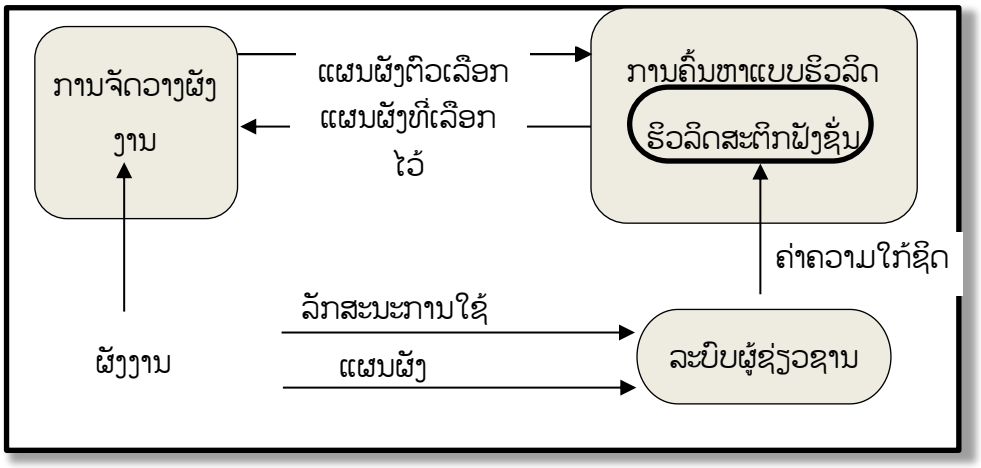
- ວິທີການຂອງການຕອບຄຳຖາມ ທີ່ຜູ້ໃຊ້ໃຫ້ຄຳຕອບຫຼາຍຢ່າງ
- ການຖາມຄືຖາມກັບຄືນ ເມື່ອຜູ້ໃຊ້ບໍ່ເຂົ້າໃຈຄຳຖາມ, ເຊັ່ນ: why, how ເປັນຕົ້ນ
- ການສະແດງຜົນທີ່ເຮັດໃຫ້ຜູ້ໃຊ້ເຂົ້າໃຈໄດ້ຈະແຈ້ງທີ່ສຸດ, ເຊັ່ນ: ການສະແດງພາບ, ຕາຕະລາງ ເປັນຕົ້ນ
- ຄວາມສາມາດໃນການເກັບຄວາມຮູ້ຈາກຜູ້ໃຊ້

ທີ່ກ່າວໄປນັ້ນເປັນສ່ວນທີ່ມີຄວາມສຳພັນໂດຍກົງກັບຜູ້ໃຊ້ລະບົບ ເພື່ອເຮັດໃຫ້ຜູ້ໃຊ້ຄຸ້ນເຄີຍກັບລະບົບໄດ້ງ່າຍ ແລະວ່ອງໄວ.

8.3. ການໃຊ້ລະບົບຜູ້ຊ່ຽວຊານສຳລັບການວາງຜັງໂຮງງານ

ຕົວຢ່າງການນຳໃຊ້ລະບົບຜູ້ຊ່ຽວຊານສຳລັບການວາງຜັງໂຮງງານມີຫຼາຍ, ແຕ່ຕົວຢ່າງໃນຫົວຂໍ້ນີ້ຈະເອົາຕົວຢ່າງທີ່ນຳສະເໜີໂດຍບຸນຈະເລີນ ແລະ ໄພລັດ (Sirinaovakul and Thajchayapong, 1994), ການເຮັດວຽກຂອງລະບົບສະແດງໃນຮູບທີ 8.7 ໂດຍເລີ່ມຕົ້ນຈາກຂະບວນການຈັດວາງເທື່ອລະພື້ນທີ່ເຮັດວຽກ ຄື: ເອົາພື້ນທີ່ເຮັດວຽກມາເຮັດການຈັດວາງເທື່ອລະພື້ນທີ່ ເພື່ອສ້າງເປັນແຜນຜັງຕົວເລືອກ (Layout Alternatives) ຫຼາຍຜັງໃຫ້ມີຫຼາຍທີ່ສຸດເທົ່າທີ່ການຈັດວາງຈະເປັນໄປໄດ້. ໃນຕອນທຳອິດຈະເລີ່ມຕົ້ນຈາກ 2 ພື້ນທີ່ເຮັດວຽກ, ຈາກນັ້ນຂະບວນການຄົ້ນຫາແບບຮີວຣິດສະຕິກ (heuristic) ຈະຄົ້ນຫາແຜນຜັງທີ່ດີຈາກແຜນຜັງຕົວເລືອກທັງໝົດຈະຖືກສ້າງຂຶ້ນ, ແລ້ວລະບົບຜູ້ຊ່ຽວຊານຈະເຮັດໜ້າທີ່ຊ່ວຍຂະບວນການການຄົ້ນຫາຮີວຣິດສະຕິກໃນການເລືອກແຜນຜັງທີ່ດີດ້ວຍການກຳນົດຄ່າຄວາມສຳພັນ (w) ໃຫ້ກັບຮີວຣິສຕິກຟັງຊັນ ເມື່ອເລືອກພື້ນທີ່ທີ່ມີການຈັດວາງທີ່ດີທີ່ສຸດຂອງ 2 ພື້ນທີ່ໄດ້ແລ້ວ, ພື້ນທີ່ທີ 3 ຈະຖືກເອົາເຂົ້າ

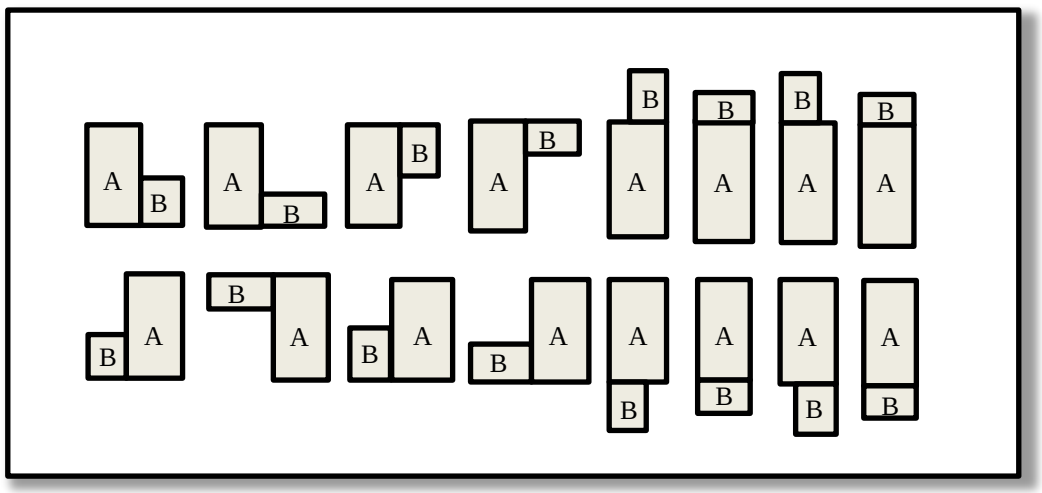
ມາຈັດວາງເພີ່ມເຂົ້າໄປໃນແຜນຜັງທີ່ໄດ້ເລືອກໄວ້ແລ້ວ ເພື່ອສ້າງແຜນຜັງຕົວເລືອກອອກມາອີກຈຳນວນໜຶ່ງ. ເຮັດການຂັດເລືອກດ້ວຍຂະບວນການຄົ້ນຫາແບບຮິວຣິດສຕິກອີກ ເມື່ອໄດ້ແຜນຜັງທີ່ດີອອກມາແລ້ວກໍໃສ່ພື້ນທີ່ທີ 4 ເຂົ້າໄປອີກແລ້ວ, ເລືອກໃໝ່ຂະບວນການຂອງການເຮັດວຽກທັງລະບົບຈະໝຸນວຽນເຊັ່ນນີ້ໄປເລື່ອຍໆຈົນບໍ່ມີພື້ນທີ່ໃໝ່ໃຫ້ວາງອີກ. ແຜນຜັງທີ່ດີທີ່ສຸດອັນສຸດທ້າຍກໍຄືຜົນໄດ້ຮັບຂອງການວາງພື້ນທີ່ເຮັດວຽກແຜນພາບການເຮັດວຽກຂອງແຕ່ລະສ່ວນຈະສະແດງໃນຮູບທີ 8.7



ຮູບທີ 8.7 ແຜນຜັງການເຮັດວຽກຂອງລະບົບທີ່ນຳສະເໜີ

8.3.1 ການຈັດວາງພື້ນທີ່ເຮັດວຽກ

ການຈັດວາງພື້ນທີ່ເຮັດວຽກຈະເລີ່ມຈາກພື້ນທີ່ເຮັດວຽກຄູ່ທຳອິດ ໂດຍເລືອກຈາກພື້ນທີ່ເຮັດວຽກຄູ່ທີ່ມີຄ່າຄວາມສຳພັນລະຫວ່າງພື້ນທີ່ເຮັດວຽກສູງສຸດຈາກຮູບທີ 8.8 ຄື A ແລະ B ຈາກນັ້ນເອົາ B ໝຸນຮອບຕົວເອງ, ແລ້ວຍ້າຍໄປທຸກມຸມຂອງ A ຈົນຄົບທຸກດ້ານ. ໃນການໝຸນໄປແຕ່ລະດ້ານຈະໄດ້ແຜນຜັງຕົວເລືອກອອກມາໜຶ່ງແບບເຮັດຊ້ຳຂະບວນການເກົ່າເຊັ່ນນີ້ໄປເລື່ອຍໆຈົນ A ແລະ B ຖືກເອົາມາປະກອບກັນທຸກດ້ານ.



ຮູບທີ 8.8 ການວາງພື້ນທີ່ເຮັດວຽກ A ແລະ B

ຈາກຮູບທີ 8.8 ຖ້າ A ແມ່ນເມເຈີ (Major) ແລະ B ແມ່ນໄມເນີ (Minor) ພື້ນທີ່ເຮັດວຽກ B ຈະໝູນຮອບໂຕເອງ, ແລ້ວໝູນຮອບພື້ນທີ່ເຮັດວຽກ A ໄດ້ແຜນຜັງອອກມາ 16 ແບບ. ໃນກໍລະນີທີ່ມີພື້ນທີ່ເຮັດວຽກ C ຖືກເລືອກເຂົ້າມາໃໝ່ແລ້ວຈະຕ້ອງວາງລົງເທິງແຜນຜັງ AB ທີ່ໄດ້ຮັບເລືອກມາແລ້ວ ກໍໃຊ້ວິທີຄ້າຍກັນຄື: ໃຫ້ C ວາງຢູ່ຕຳກັບດ້ານໃດດ້ານໜຶ່ງຂອງພື້ນທີ່ເຮັດວຽກ A ຫຼື B ຈາກນັ້ນກໍເຮັດການໝູນ C ຮອບຕົວເອງ, ແລ້ວໝູນຮອບ B, ຈາກນັ້ນກໍໝູນຮອບ Aຈາກຜົນຂອງການໝູນທັງໝົດໃຫ້ເລືອກສະເພາະແຜນຜັງທີ່ບໍ່ມີພື້ນທີ່ເຮັດວຽກທີ່ວາງຊ້ອນທັບກັນແຜນຜັງທີ່ໄດ້ອອກມາຈະມີຫຼວງຫຼາຍ ຕາຕະລາງທີ 8.1 ເປັນຈຳນວນແຜນຜັງທີ່ໄດ້ກັບຈຳນວນພື້ນທີ່ເຮັດວຽກ.

ຕາຕະລາງ 8.1 ຈຳນວນແຜນຜັງທີ່ສ້າງຂຶ້ນຕໍ່ຈຳນວນພື້ນທີ່ເຮັດວຽກ

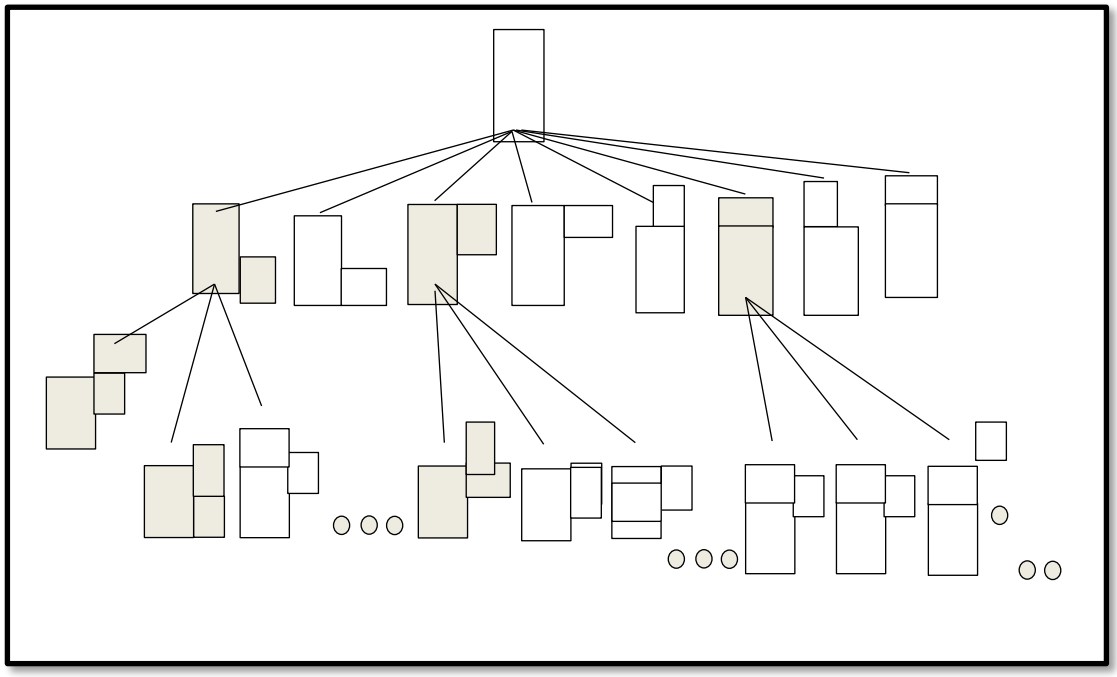
ຈຳນວນໜ່ວຍງານ	ຈຳນວນແຜນຜັງທີ່ໄດ້
2	32
3	1,792
4	143,360
5	14,909,440
6	1,908,408,320

8.3.2 ການຄົ້ນຫາແບບຮິວຣິດສະຕິກ (Heuristic Search)

ການຄົ້ນຫາຈະໃຊ້ການຄົ້ນຫາແບບບີມ (Beam Search) ໂດຍເລີ່ມຕົ້ນຈາກການຄັດເລືອກພື້ນທີ່ເຮັດວຽກທີ່ມີຄ່າຄວາມສຳພັນລະຫວ່າງພື້ນທີ່ເຮັດວຽກສູງສຸດ, ແລ້ວໃຊ້ອານກໍລິທຶມ (algorithm) ການຈັດວາງຮູບສ້າງແຜນຜັງຕົວເລືອກພ້ອມຄຳນວນຄ່າໃຊ້ຈ່າຍຂອງແຕ່ລະແຜນຜັງໂດຍໃຊ້ຮິວຣິດສະຕິກຟັງຊັນ (heuristic function) ຈາກຄ່າໃຊ້ຈ່າຍນີ້ແຜນຜັງທີ່ມີຄ່າໃຊ້ຈ່າຍນ້ອຍກໍ່ຈະຖືກເລືອກອອກມາຈຳນວນໜຶ່ງ. ຈາກນັ້ນລະບົບກໍ່ຈະເລືອກພື້ນທີ່ເຮັດວຽກ ໃໝ່ທີ່ຈະນຳເຂົ້າມາວາງຕໍ່ໄປ ໂດຍພິຈາລະນາຈາກຄ່າຄວາມສຳພັນສູງສຸດພື້ນທີ່ເຮັດວຽກໃໝ່ຈະຖືກນຳມາວາງເທິງແຜນຜັງທີ່ເລືອກມາແລ້ວຈາກຂັ້ນຕອນທີ່ຜ່ານມາ ໂດຍອາໄສຂະບວນການຈັດວາງຮູບແບບຄຳນວນຄ່າໃຊ້ຈ່າຍຂອງແຕ່ລະແຜນຜັງເພື່ອນຳມາພິຈາລະນາໃນການເລືອກອີກ. ຂະບວນການຂອງການຄົ້ນຫານີ້ຈະເຮັດຊ້ຳເຊັ່ນນີ້ໄປເລື້ອຍໆ ຈົນກວ່າທຸກພື້ນທີ່ເຮັດວຽກຖືກວາງທັງໝົດ ແຜນຜັງຊຸດສຸດທ້າຍທີ່ຖືກເລືອກອອກມາຈະເປັນແຜນຜັງທີ່ຕ້ອງການ. ສະຫຼຸບແລ້ວຂະບວນການວາງແຜນຜັງຂອງລະບົບທີ່ນຳສະເໜີຈະເປັນດັ່ງນີ້:

1. ວາງພື້ນທີ່ເຮັດວຽກໂດຍຂະບວນການວາງພື້ນທີ່ເຮັດວຽກຜົນທີ່ໄດ້ຄືແຜນຜັງຕົວເລືອກ (Alternative layouts)
2. ຈາກຜົນທີ່ໄດ້ຈາກຂັ້ນຕອນທີ 1, ເລືອກແຜນຜັງທີ່ມີຄ່າໃຊ້ຈ່າຍຕ່ຳອອກມາຈຳນວນໜຶ່ງ
3. ຖ້າຍັງມີພື້ນທີ່ເຮັດວຽກທີ່ຕ້ອງວາງ, ໃຫ້ເລືອກພື້ນທີ່ເຮັດວຽກໃໝ່ທີ່ຈະຈັດວາງແລ້ວກັບໄປເຮັດຂັ້ນຕອນທີ 1. ໃນຮູບທີ 8.9 ສະແດງຂັ້ນຕອນການເຮັດວຽກຕາມຂໍ້ສະຫຼຸບຂ້າງຕົ້ນ, ໂດຍເລີ່ມຈາກສ່ວນເທິງສຸດຂອງໂຄງສ້າງຕົ້ນໄມ້ທີ່ໄດ້ຈາກການເລືອກພື້ນທີ່ເຮັດວຽກຄູ່ທີ່ມີຄວາມໃກ້ຊິດສູງສຸດແລ້ວ

ນຳພື້ນທີ່ເຮັດວຽກທີ່ມີພື້ນທີ່ຫຼາຍສຸດເປັນພື້ນທີ່ເຮັດວຽກເລີ່ມຕົ້ນ. ຈາກນັ້ນກໍເຮັດການຈັດວາງພື້ນທີ່ເຮັດວຽກທີ2 ຈະໄດ້ຮູບແບບຕ່າງໆຂອງການຈັດວາງພື້ນທີ່ເຮັດວຽກອອກມາຫຼວງຫຼາຍ (ໃນຮູບສະແດງໄວ້ແຕ່8ແຜນຜັງເທົ່ານັ້ນ) ແລ້ວເຮັດການເລືອກແຜນຜັງທີ່ດີທີ່ສຸດອອກມາ 3 ຜັງ (ບີມວິດ= 3) ເຊິ່ງກໍຄືແຜນຜັງທີ່ແລເງົາໄວ້ໃນລະດັບທີ 1 ຂອງໂຄງສ້າງຕົ້ນໄມ້ ແລະ ຈາກແຜນຜັງທີ່ເລືອກໄວ້ທັງສາມກໍເຮັດການວາງພື້ນທີ່ເຮັດວຽກທີ3ໃນແຕ່ລະພື້ນທີ່ເຮັດວຽກທີ່ວາງໄວ້ຈະໄດ້ແຜນຜັງລູກອອກມາອີກຈຳຫຼາຍ(ໃນທີ່ນີ້ສະແດງໄວ້ແຕ່3ແຜນຜັງ)ຕາມທີ່ສະແດງໄວ້ໃນລະດັບທີ2ຂອງໂຄງສ້າງຕົ້ນໄມ້ແລະຈາກທັງໝົດໃຫ້ເລືອກແຜນຜັງທີ່ດີທີ່ສຸດອອກມາ3 ແຜນຜັງຕາມທີ່ແລເງົາແລ້ວເຮັດການໃສ່ພື້ນທີ່ໃຫ້ວາງອີກຜົນການເລືອກແຜນຜັງເທື່ອສຸດທ້າຍກໍຄືແຜນຜັງທີ່ເປັນຜົນຮັບ.



ຮູບທີ 8.9 ຂັ້ນຕອນການຈັດວາງ ແລະ ການຄົ້ນຫາແບບບີມ

8.3.3 ລະບົບຜູ້ຊ່ຽວຊານ

ຖານຄວາມຮູ້ໃນລະບົບຜູ້ຊ່ຽວຊານຄືກົດທີ່ສະແດງຂະບວນການກຳນົດຄ່າຄວາມສຳພັນຂອງພື້ນທີ່ເຮັດວຽກ(ຫຼືສຳປະສິດຄວາມແຂງຕົງ)ໂດຍພິຈາລະນາຈາກໄລຍະການໃຊ້ງານງານຂອງພື້ນທີ່ການເຮັດວຽກນັ້ນໆຖ້າຈະຕ້ອງວາງພື້ນທີ່ເຮັດວຽກໄວ້ຕິດກັນຄ່າຄວາມສຳພັນນີ້ຈະຕ້ອງມີຄ່າສູງຖ້າຈະຕ້ອງພື້ນທີ່ເຮັດວຽກຫ່າງກັນຄ່າສຳປະສິດນີ້ຈະຕ້ອງມີຄ່າຕ່ຳໃນບາງຄັ້ງການກຳນົດນີ້ອາດຈະຕ້ອງກຳນົດພິເສດອອກໄປ ເຊັ່ນ:ໃນໂຮງງານທີ່ມີພື້ນທີ່ເຮັດວຽກເຊື່ອມແລະພື້ນທີ່ເຮັດວຽກທຳຄວາມສະອາດຜິວວັດສະດຸເມື່ອພິຈາລະນາຢ່າງຄ່າວໆແລ້ວພື້ນທີ່ເຮັດວຽກທັງສອງນີ້ຄວນຢູ່ຕິດກັນເພາະເຊື່ອມແລ້ວຈະໄດ້ເຮັດຄວາມສະອາດເລີຍແຕ່ໃນການວາງແຜນຈິງອາດເຮັດເຊັ່ນນີ້ບໍ່ໄດ້ເພາະພື້ນທີ່ສຳລັບເຮັດຄວາມສະອາດຜິວວັດ

ສະດຸຈະມີສານເຄມີທີ່ກໍ່ໃຫ້ເກີດການລຸກໄໝ້ຈາກແປ່ວໄຟຂອງການເຊື່ອມໄດ້ດັ່ງນັ້ນພື້ນທີ່ເຮັດວຽກທັງສອງ ບໍ່ຄວນຢູ່ຕິດກັນເປັນຕົ້ນຮູບແບບຂອງການຂຽນກົດສຳລັບຜູ້ຊ່ຽວຊານເປັນດັ່ງນີ້:

LABEL: IF CONITION

IHEN WEIGHT=VALUE.

ໃນກົດຂອງIF_THANເມືອLABELເປັນຊື່ກຳກັບຂອງກົດCONDITIONສຳພັນຂອງພື້ນທີ່ ເຮັດວຽກWEIGHTເປັນຄ່າສະເພາະທີ່ຈະຕ້ອງມີສະເໜີໃນກົດທຸກຂໍ້ແລະVALUEເປັນຄ່າທີ່ ຜູ້ຊ່ຽວກຳນົດພາຍໃຕ້ເງືອນໄຂຂອງCONDITION

ຕົວຢ່າງການເຮັດວຽກຂອງລະບົບ

ໃນໂຮງງານຂະນາດນ້ອຍແຫ່ງໜຶ່ງມີພື້ນທີ່ເຮັດວຽກທີ່ເຮັດໜ້າທີ່ຕ່າງໆດັ່ງຕໍ່ໄປນີ້

1.ຫ້ອງຂຶ້ນຮູບໂລຫະ

2.ຫ້ອງເຊື່ອມ

3.ຫ້ອງເຮັດຄວາມສະອາດດ້ວຍສານເຄມີ

4.ຫ້ອງເກັບເຄື່ອງ

ຈາກຂໍ້ມູນເກົ່າທີ່ເຄີຍເຮັດການບັນທຶກໄວ້ກ່ຽວກັບຂໍ້ມູນຕ່າງໆຂອງການໄຫຼຂອງວັດສະດຸໄປຕາມ ຫ້ອງແລະຂະນາດຂອງຫ້ອງເປັນດັ່ງນີ້

ຕາຕະລາງທີ 8.2 ຂໍ້ມູນຂອງພື້ນທີ່ເຮັດວຽກຕ່າງໆ

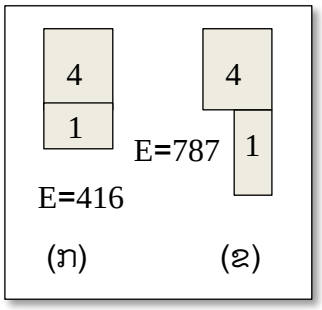
	ຫ້ອງຂຶ້ນຮູບ	ຫ້ອງເຊື່ອມ	ຫ້ອງທຳຄວາມສະອາດ	ຫ້ອງເກັບເຄື່ອງ	ຂະໜາດ
ຫ້ອງຂຶ້ນຮູບ 1	-	20	10	20	2x1
ຫ້ອງເຊື່ອມ 3	-	-	20	18	2x1
ຫ້ອງທຳຄວາມສະອາດ 3	-	-	-	15	1x1
ຫ້ອງເກັບເຄື່ອງ 4	-	-	-	-	2x2

ລັກສະນະຂອງຫ້ອງຕ່າງໆເປັນດັ່ງນີ້ຫ້ອງເຮັດຄວາມສະອາດແລະຫ້ອງເຊື່ອມຈະຢູ່ຕິດກັນບໍ່ໄດ້ ວັດສະດຸໄຫຼວຽນລະຫ່ວ່າງຫ້ອງຂຶ້ນຮູບ,ຫ້ອງເຊື່ອມ,ຫ້ອງເກັບຂອງແລະຫ້ອງເຮັດຄວາມສະອາດມີ ລັກສະນະໜັກຫຼາຍແລະເສຍຫາຍງ່າຍສຳລັບວັດສະດຸທີ່ໄຫຼວຽນລະຫ່ວ່າງຫ້ອງເຊື່ອມແລະຫ້ອງເຮັດຄວາມ ສະອາດສ່ຽງຕໍ່ຄວາມເສຍຫາຍງ່າຍ

ໂດຍການໃຫ້ຄຳປຶກສາຂອງລະບົບຜູ້ຊ່ຽວຊານຈະໄດ້ຄ່າຄວາມສຳພັນເປັນດັ່ງນີ້

W=5.1, w=8.97, w= 9.26, w=0,w= 3 ແລະ w=8.53

ຄ່າທີ່ຫຼາຍທີ່ສຸດຂອງw ຄື w14ດັ່ງນັ້ນໃຫ້ເລືອກພື້ນທີ່ເຮັດວຽກ1ແລະ4ມາເຮັດການຈັດວາງແລະເລືອກ ແຜນຜັງທີ່ດີທີ່ສຸດອອກມາ2ຜັງດັ່ງສະແດງໃນຮູບທີ 8.10



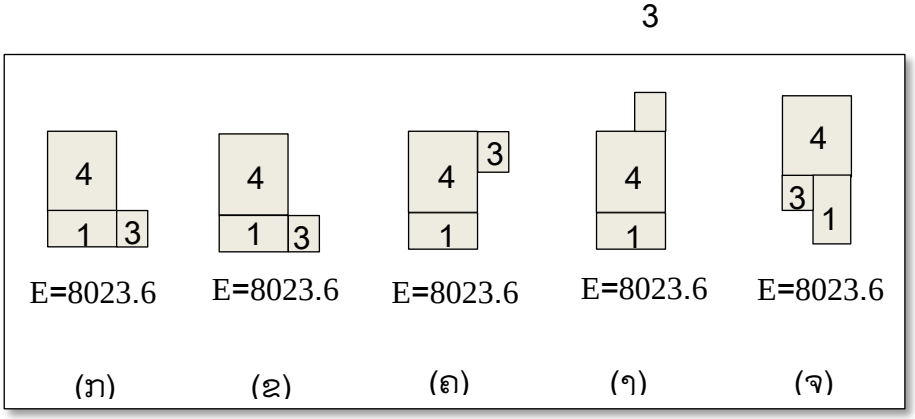
ຮູບທີ 8.10 ຜົນຮັບຂອງການວາງພື້ນທີ່ເຮັດວຽກ 1 ແລະ 4

ການເລືອກພື້ນທີ່ເຮັດວຽກທີ່ຈະວາງຕໍ່ໄປພິຈາລະນາດັ່ງນີ້

-ຄ່າລວມຂອງ w ຂອງພື້ນທີ່ເຮັດວຽກ2ກັບພື້ນທີ່ເຮັດວຽກທີ່ຈັດວາງແລ້ວໄດ້ແກ່ພື້ນທີ່ເຮັດວຽກ 2 ກັບພື້ນທີ່ເຮັດວຽກ1ແລະພື້ນທີ່ເຮັດວຽກ 2 ກັບພື້ນທີ່ເຮັດວຽກ4ເທົ່າກັບ w+w=5.1+3=8.1

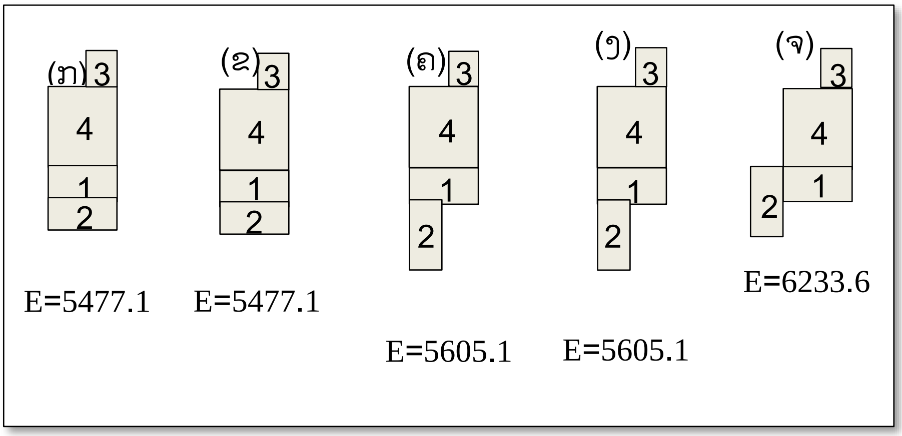
-ຄ່າລວມຂອງwຂອອງພື້ນທີ່ເຮັດວຽກ 3 ກັບພື້ນທີ່ທີ່ຈັດວາງແລ້ວໄດ້ແກ່ພື້ນທີ່ເຮັດວຽກ 3 ກັບພື້ນທີ່ເຮັດ ວຽກ 1 ແລະພື້ນທີ່ເຮັດວຽກ 3ກັບພື້ນທີ່ເຮັດວຽກ 4 ເທົ່າກັບ w+w=8.97+8.53=112.5

ຄ່າຄວາມສຳຄັນລວມຂອງພື້ນທີເຮັດວຽກ 3 ສູງກ່ວາຂອງພື້ນທີ່ເຮັດວຽກ 2 ດັ່ງນັ້ນບ່ອນເຮັດ 3ຄືພື້ນທີ່ ເຮັດຕໍ່ໄປທີ່ຈະຈັດວາງເຊິ່ງຜົນຕາມຮູບທີ 8.11



ຮູບທີ 8.11ຜົນຮັບຂອງການເລືອກແຜນຜັງໃສ່ພື້ນທີ່ເຮັດວຽກທີ3

ຈາກນັ້ນໃສ່ພື້ນທີ່ເຮັດວຽກທີ2ເຂົ້າໄປລັບຊິໄດ້ຕາມຮູບ 8.12



ຮູບທີ 8.12 ຜົນຮັບຂອງການວາງໃສ່ພື້ນທີ່ເຮັດວຽກ

ຜົນຂອງການວາງພື້ນທີ່ເຮັດວຽກຄືຮູບທີ 8.12 (ກ) ຫຼື (ຂ) ເຊິ່ງເປັນພທ້ນທີ່ເຮັດວຽກທີ່ມີຄ່າ E ຕ່ຳທີ່ສຸດ ແລະຖືວ່າການເຮັດພື້ນທີ່ເຮັດວຽກທີ່ເປັນຄຳຕອບ

ເອກະສານອ້າງອີງ

1. Mitchell, T. M. (1997). *Machine Learning.* New York: McGraw-Hill.
2. N.Sebe, Ira Chen, Ashtosh Garg and Thomas .S Huang. (2005). *Machine Learning in Computer Vision.* Dordrecht: Springer.
3. Norris, D. J. (2017). *Begining Artificial Intelligence with the Respbery Pi.* Barrington: Apress.
4. Rich, E. a. (1991). *Artificial Intelligence.* New York: McGraw-Hill.
5. Urwin, R. (2016). *Artificial Intelligence.* London: ARCTURUS.
6. ກິດຕິກຸນ, ບ. (2005). *Artificial Intelligence.* ກຸງເທບ.
7. ສີຣິເມົາອະກຸນ, ບ. (2014). *ປັນຍາປະດິດ.* ກຸງເທບ: ສຳນັກພິມທອບ.
8. ພັກດີວັດທະນະກຸນ, ກ. (2009). *ຄຳພີລະບົບສະໜັບສະໜູນການຕັດສິນໃຈ ແລະ ລະບົບຜູ້ຊ່ຽວຊານ.* ກຸງເທບ: ເຄທີພີ.