REFERENCE:

AgentSheets, Inc. 2006, *AgentSheet*, Retrieved 20th Nov. 2006 from http://www.agentsheets.com/index.html

Arnold, G. & Valk, V. 1992, 'Establishment, colonization and persistence', in *Plant succession: Theory and Prediction, Population and Community Biology Series 11,* eds D.C.Glenn-Lewin, R.K. Peet & T.T. Veblen, Chapman and Hall, pp. 60 - 103.

Ashton, D.H. 1979, 'Seed harvesting by ants in forests of Eucalyptus regnans F. Muell. in central Victoria'. *Australia Journal of Ecology*, vol. 4, pp. 265 - 277.

Bampfylde, C.J., Brown, N.D., Gavaghan, D.J. & Maini, P.K. 2005. Modelling rain forest diversity: The role of competition, *Ecological Modelling*, vol. 188, pp. 253 - 278.

Baveco, J.M. & Lingeman, R. 1992, 'An object-oriented tool for individual-oriented simulation: host-parasitoide system application', *Ecological Modelling*, vol. 61, pp. 267 - 286.

Bell, F.G., Bullock, S.E.T., Halbich, T.F.J. & Lindsay. P. 2001, Environmental impacts associated with an abandoned mine in Witbank Coalfield, South Africa, *Coal Geology*, vol. 45, pp. 195 - 216.

Bell, L.C. 2001, 'Establishment of native ecosystems after mining-Australia experience across diverse biogeographic zones', *Ecological Engineering*, vol. 17, pp. 179 - 186.

Bellairs, S.M. & Bell, D.T. 1993, 'Seed stores for restoration of species rich shrubland vegetation following mining in Western Australia', *Restoration Ecology*, vol. 1, pp. 231 - 240.

Bellairs, S.M. 1996, 'Of what value is the soil seed bank for minesite rehabilitation in Queensland', in *Proceedings of the 2nd National Workshop on Native Seed Biology for Revegetation,* eds. S.M.Bellairs and L.C.Bell, Newcastle, pp. 149 - 154

Boland, D.J., Booker, M.I.H., Chippendale, G.M., Hall, N., Johnston, R.D., Kleinig, D.A. & Turner, J. 1984, *Forest Trees of Australia*, 4th edn. Nelson, Melbourne.

Botkin, D.B. 1979, 'A grandfather clock down the staircase: Stability and disturbance in natural ecosystems', In *Forests: Fresh Perspectives from Ecosystem Analysis,* eds R.H. Waring, Oregon State University Press, Corvallis, pp. 1 - 10.

Botkin, D.B., Janak, J.F. & Wallis, J.R. 1972, 'Some ecological consequences of a computer model of forest growth', *Journal of Ecology,* vol. 60, pp. 849 - 873.

Bousquet, F., Bakam, I., Proton, H., Le Page, C. 1998, 'Cormas : Common-pool resources and multi-agent systems', *Lecture notes in computer science*, Proceedings of the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems: Tasks and Methods in Applied Artificial Intelligence, Springer, Berlin. vol. 1416, pp. 826 - 837

Bowen, G.D. 1956, 'Nodulation of legumes indigenous to Queensland', *Queensland Journal of Agricultural Science*, vol.12, pp.47 - 60.

Bradshaw, A.D. & Chadwick. 1980, *The Restoration of Land: the Ecology and Reclamation of Derelict and Degraded land,* University of California Press, Berkeley, CA.

Bradshaw, A.D., Marrs, R.H. & Roberts, R.D. 1982, 'Succession in Ecology of Quarries', in *The importance of Natural Vegetation,* eds B.N.K. Davis, *Inst. Terrestrial Ecol. Symp.,* Vol. 11, pp. 47 - 52.

Bradshaw, A.D. 1983, 'The reconstruction of ecosystems', *Journal of Applied Ecology*, vol. 20, pp. 1 - 17.

Bradshaw, A. D. 1984, 'Land restoration: now and in the future', *Proc. Royal Soc*, B223, pp. 1 - 13.

Bradshaw, A.D. 1993, 'Restoration ecology as science', *Restoration Ecology*, vol. 1, pp. 71 - 73.

Bravo-Oviedo A, Sterba H, del Rio M, & Bravo, F., 2006, Competition-induced mortality for Mediterranean Pinus pinaster Ait. and P-sylvestris L. *Forest Ecology and Management,* 222 (1-3): 88-98

Brooks, D.R. & Yeates, D.J. 1980, 'Ecosystem development in frontal dunes after mining', *Landline,* vol. 4, pp. 2 - 3

Brooks, D.R. 1981, 'The planning and logistics of mineral sands rehabilitation', in AMIC Environmental Workshop, Canberra, Australia, pp. 80 - 92.

Bruns, G., Mössinger, P., Polani, D., Schmitt, R., Spalt, R., Uthmann, T. & Weber, S. 2003, XRaptor, 'A simulation environment for continuous virtual multi-agent systems, user manual', Retrieved at 7th July, 2006, from http://www.informatik.uni-mainz.de/~polani/XRaptor/

Burns, M.W. 1987, 'Delineation of Physical and Chemical Factors Affecting Native Tree Establishment on Reformed Open-cut Coal Mine Sites within the Hunter Valley', Master Degree, University of New England, Armidale.

Burrows, F.J. 1986, 'The effect of taproot length and soil disturbance on the growth of Angophora costata seedlings planted on dunes restored after mineral sand mining', *Reclam. and Reveg. Res.*, vol. 4, pp. 167 - 182.

Cairns, J.J. 1991, 'The status of the theoretical and applied science of restoration ecology', *The environmental professional*, vol. 13, pp. 186 - 194.

Camporeale, C. & Ridolfi, L. 2006, ' Riparian vegetation distribution induced by river flow vriability: a stochastic approach', *Water Resources Research*,42(10)

Cattelino, P.L., Noble, I.R., Slatyer, R.O. & Kessell, S.R. 1979, 'Predicting the multiple pathways of plant succession', *Environ. Mgmt.*, vol. 3, pp.41 - 50.

Chapman, G.P. 1992. *Desertified Grassland,* Academy Press, London.

Chaulya, S.K., Singh, R.S.,Chakraborth, M.K.& Dhar, B.B 1999, Numerical modelling of biostabilisation for a coal mine overburden dump slope, *Ecological Modelling*, vol. 114 (2-3), pp. 275 - 86.

Chertov, O.G., Komarov, A.S., & Tsiplianovsky, A.M. 1999, 'A combined simulation model of Scots pine, Norway spruce and Silver birch ecosystems in the European boreal zone', *Forest Ecology and Management*, vol. 116, pp. 189 - 206.

Chesterfield, C.J.&. Parsons, R.F., 1985, 'Regeneration of three tree species in arid south-eastern Australia', *Australian Journal of Botany*, vol. 33, pp.715 - 732.

Choler, P, Michalet, R. & Callawy, R.M. 2001. Facilitation and competition on gradient in alpine plant communities. *Ecology*. vol. 82. pp. 3295 - 3308.

Clements, F.E. 1916, *Plant succession: an analysis of the development of vegetation,* Carnegie Inst., Wash. Publ.

Clements, F.E. 1936, 'Nature and structure of the climax', *Journal of Ecology*, vol. 24, pp. 252 - 284.

Colella, V.S., Klopfer, E. & Resnick, M. 2001, *Adventures in Modeling: Exploring complex, Dynamic Systems with Starlogo*, Teachers College Press.

Collier, N. 2003, 'RePast: An RePast: An Extensible Framework for Agent Simulation', Social Science Research Computing, University of Chicago, Chicago, IL 60637. Retrieved 15th Nov. 2005 from http://repast.sourceforge.net/docs/repast_intro_final.doc

Collins, S.L., Glenn, S.M.& Gibson, D.J. 1995, 'Experimental analysis of intermediate disturbance and initial floristic composition: decoupling cause and effect', *Ecology*, vol. 76, pp. 486-492.

Connell, J.H. & Slatyer, R.O. 1977, 'Mechanisms of succession in natural communities and their role in community stability and organization', *American Naturalist*, vol. 111, pp. 1119 - 1144.

Connell, J.H., 1987, Diversity in Tropical Rain Forests and Coral Reefs, *Science*, Vol. 199, Issue 4335, pp. 1302 - 1310.

Connell, J.H., Noble, I.R., & Slatyer, R.O. 1987, 'On the mechanisms producing successional change', *Oikos*, vol. 50, pp. 136 - 137.

Cooke, J.A. 1999, 'Mining', in *Ecosystems of the World 16: Ecosystems of Disturbed Ground*, eds L.R.Walker, ELSEVIER.

Cooke, J. 2002, 'Ecological restoration of land with particular reference of the mining of metals and industrial minerals: A review of theory and practice', *Environment Review*, vol. 10, pp. 41 - 71.

Cremer, K.W. 1965, 'Emergency of *Eucalyptus regnans* from buried seed', *Australia Forest*, vol. 29, pp. 119 - 124.

Cremer, K.W. 1966. 'Dissemination of seed from *Eucalyptus regnans*.' *Australia Forest*, 30, 33 - 37.

Cremer, K.W. (eds) 1990, *Trees for Rural Australia*, Inkata Press, Sydney, NSW.

Croft, J.B. 1979, 'Report on the Botany, Wildlife and Ecology of Leard State Forest', a report, Newcastle.

Cunningham G.M. 1966, 'A survey of tree planting and tree growth rates in the Condobolin district', *J. Soil Conserv. Serv.*, N.S.W. vol. 22, pp. 174 - 89

Cunningham, G.M., Mulham, W.E., Milthorpe, P.L. & Leigh, J.H. 1981, *Plants of Western New South Wales*, NSW Government Printer: Sydney.

Cunningham, G.M., Mulham, W.E., Milthorpe, P.L., & Leigh, J.H. 1992, *Plants of Western New South Wales,* Inkata Press.

Dahl, N.W., & Mulligan, D.R. 1996, 'Environmental management at Weipa -Bauxite mining in the tropical north', in *Environmental Management in the Australia Mineral and Energy Industries: Principles and Practices,* eds D.R. Mulligan, UNSW Press, pp. 383 - 438.

Daily, G.C. 1995, 'Restoring value to the worlds degraded lands', *Science*, vol. 269, pp. 350 - 354.

Denham, R.J. 1989, 'Recolonisation of coal mine overburden by *Callitris glaucophylla* Thompson and Johnson at Boggabri, N.S.W.', Bachelor of Nature Resource, University of New England, Armidale.

DEST. 1996, 'Australia: State of the Environment 1996', State of the Environment Advisory Council, Department of the Environment, Sport and Territories, Canberra.

Drake, J.A. 1990, 'The mechanics of community assembly and succession', *J. Theor. Biol.*, vol. 147, pp. 213 - 233.

Duggin, J.A., Gouvernet, J.M., Fosdick, M. & Watts, T.R.G., 1982, ' Rehabilitation studies for the proposed Boggabri open-cut coal mine Amax: BHP Boggabri Coal Project', Natural Resources Project Report P.R. No. 19, Department of Ecosystem Management, University of New England, Armidale, NSW.

Duggin, J.A. 1992, 'Australian coal association research program (ACARP)', proposals for 1992 - 1993. Unpublished. University of New England.

Durrett, R.& Levin, S. 1994. The importance of being discrete (and spatial). *Theoretical Population Biology*, vol. 46, pp. 363 - 394

EMBYR, 2007, Oak Ridge National Laboratory, U.S. Department of Energy, last retrieved at 2nd Feb 2007 from http://www.ornl.gov/

Eckel, B. 2002, *Thinking in Java,* 3rd edition, Prentice-Hall. http://www.cs.hut.fi/Docs/Eckel/

Egler, F.E. 1954, 'Vegetation science concepts. I. Initial floristic composition, a factor in old field vegetation development', *Vegetation*, vol 4.

Ehrenfeld, J.G. 2000, 'Defining the limits of restoration: the need for realistic goals', *Restoration Ecology*, vol. 8, pp. 2 - 9.

ESRI, 2006, 'ESRI GIS and Mapping Software', Last retrieve 16th Nov. 2006, from http://www.esri.com/

Ezoe, H. & Nakamura, S., 2006, 'Size distribution and spatial autocorrelation of subpopulations in a size structured metapopulation model', *Ecological Modelling*, vol. 198, issue 3 - 4, pp. 293 - 300

Faliñska, K. (eds) 1998, *Plant Population Biology and Vegetation Processes*, W. Szafer Institute of Botany, Polish Academy of Sciences, Krakow, Poland.

Falster DS & Westoby M, 2003, Plant height and evolutionary games, *Trends in Ecology & Evolution,* 18 (7): 337-343

FAO. 2000. 'On Definitions of Forest and Forest Change, Forest Resources Assessment Programme Working Paper 33. FAO, Rome, Italy.

Farrell, T.J., & Kratzing, D.C. 1996, 'Environmental effects', in *Environmental Management in the Australian Minerals and Energy Industries: Principles and Practices,* eds. D.R. Mulligan, UNSW Press, Sydney, Australia, pp. 14 - 45.

Ferguson, K.D. & Erickson, P.M. 1988, 'Pre-mine prediction of acid mine drainage', in *Dredged Material and Mine Tailings*, eds. W. Salomons & U. Forstner, Springer-Verlag Berlin Heidelberg.

Fishwick, P.A. & Sanderson, J.G. 1996, 'A Multimodeling Basis for Across-Trophic-Level Ecosystem Modeling: The Florida Everglades Example'. Retrieved 12th Dec. 2003 from http://www.cise.ufl.edu/~fishwick/tr/tr96-038.html.

Flores-Mendez, R.A. 1999, 'Towards a Standardization of Multi-Agent System Frameworks', retrieved 12th Dec. 2005, from http://www.acm.org/crossroads/xrds5-4/multiagent.html

Fox, B.J. 1990, 'Two hundred years of disturbance: how has it aided our understanding of succession in Australia?', in *Proceedings of the Ecological Society of Australia*, 1990. Vol. 16, pp. 521 - 529.

Franklin, S. & Graesser, A. 1997, 'Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents', in *Intelligent Agent III: Agent Theories, Architectures, and Languages*, *Lecture Notes in Artificial Intelligence 1193,* eds. J.-P. Muller, M.J. Wooldridge & J.P. Jennings, Springer, Berlin, Heidelberg, pp, 21 - 35

GeoScience Australia, 2006, Australia Government Geoscience Australia, Retrieved 10th, November 2005 from http://www.ga.gov.au/

Getzin, S., Dean, C., He, FL., Trofymow, JA., Wiegand, K. & Wiegand, T. 2006, "Spatial pattern and competition of tree species in a Douglas-fir chronosequence on Vancouver Island', *Ecography*, 29(5), pp. 671 - 682

Gilbert, N. & Troitzsch, K. 1999, *Simulation for the Social Scientist*, Open University Press, Buckingham.

Gleason, H.A. 1926, 'The individualistic concept of the plant association', *Bulletin of Torrey Botanical Club*, 53.

Gleason, H.A. 1939, 'The individualistic concept of the plant association', *Amer. Mid. Nat.*, vol. 21, pp. 92 - 110.

Glen-Lewin, D.C. 1980, 'The individualistic nature of plant community development', *Vegetation*, vol. 43, pp. 141 - 146.

Glen-Lewin, D.C. & van der Maarel, E. 1992, 'Vegetation dynamics', in *Plant succession: theory and prediction,* eds. D.C. Glen-Lewin, R.K. Peet & T.T. Veblen, Chapman & Hall, London.

Glen-Lewin, D.C., Peet, R.K. & Veblen, T.T. 1992, *Plant Succession: theory and prediction,* Chapman & Hall, London.

Gouvernet, J.M. 1981, 'Rehabilitation Research for the Proposed Boggabri Open-Cut Coal Mine', Bachelor Nature Resource Honours thesis, University of New England, Armidale, Australia.

Governor, T.R. & Secretary, J.M.S. 1998, *Coal Mine Drainage Prediction and Pollution Prevention in Pennsylvania,* The Pennsylvania Department of Environmental Protection, Pennsylvania.

Grant, C.D. & Koch, J.M. 1997, 'Ecological aspects of soil seed banks in relation to bauxite mining: (II) Twelve year-old rehabilitated mines', *Australian Journal of Ecology*, vol. 22, pp. 177 - 184.

Grant, C.D., Duggin, J.A., Meek, I.K., & Lord, M. 2001, 'Endpoint criteria and successional pathways for manganese mining rehabilitation on Groote Eylandt, Northern Territory', In 26th Annual Minerals Council of Australia, Environmental Workshop - Back to the Future, Adelaide, pp. 129 - 142.

Grant, C.D. 2006, 'State-and-Transition Successional Model for Bauxite Mining Rehabilitation in the Jarrah Forest of Western Australia', *Restoration Ecology*, vol. 14, pp. 28 - 37.

Grigg, A.H. 1987, 'Tree and Shrub Recolonisation of Coal Mine Overburden- Boggabri, N.S.W.', Bachelor of Natural Resources Honours Thesis, University of New England, Armidale, Australia.

Grimm, V. 1999, 'Ten years of individual - based modelling in ecology: what have we learned and what could we learn in the future?', *Eco. Modelling*, vol. 115, pp. 129 - 148.

Gross, K.L. 1987, 'Mechanisms of colonization and species persistence in plant communities', In *Restoration Ecology: Theory and Practice*, W.R. Jordan, M. Gilpin, and J. Aber (eds), Cambridge University Press, London. pp. 173 - 188,

Group, Asher Joel Media Group (eds) 1984, *Australian mining technology,* Sydney.

Hall, N.H., Boden, R.W., Christian, C.S., Condon, R.W., Dale, F.A., Hart, A.J., Leigh, J.H., Marshall, J.K., McArthur, A.G., Russell, V. & Turnbull, J.W., 1972, *The Use of Trees and Shrubs in the Dry Country of Australia.* Australian Government Publishing Service, Canberra.

Hancock, G.R., Grabham, M.K., Martin, P., Evans, K.G. & Bollhöfer, A. 2006. 'A methodology for the assessment of rehabilitation success of post mining landscapes - sediment and radionuclide transport at the former Nabarlek uranium mine, Northern territory, Australia', *Science of Total Environment*, vol. 354, pp. 103 - 119.

Hancock, G.R. & Turley, E. 2006, 'Evaluation of proposed waste rock dump designs using the SIBERIA erosion model', *Environ. Geol.*, vol. 49, pp. 765 - 779.

Hannan, J.C. 1981, 'The use of topsoil in coal mine rehabilitation programs. In Environmental Controls for Coal Mining', in *First National Seminar*, eds J.C. Hannan, The Australian Coal Association and the Earth Resources Foundation.

Hannan, J.C. 1995, *Mine Rehabilitation. A Handbook for the Coal Mining Industry,* 2nd edn., New South Wales Coal Association, Sydney

Harden, G.J. (eds) 2000, *Flora of New South Wales*, UNSW University Press, Kensington, N.S.W.

Harrington, G.N., Oxiey, R.E., & Tongway. 1979, 'The effects of European settlement and domestic livestock on the biological system in the poplar box (*Eucalyptus populnea*) lands', *Aust. Range. J.*, vol. 1, pp. 271 - 279.

Hartvigsen, G. & Levin, S. 1997, 'Evolution and spatial structure interact to influence plant - herbivore population and community dynamics', in Proceedings of the Royal Society of London B Biological Sciences, Vol. 264, pp. 1677 - 1685.

Hatton, T.J. & West, N.E. 1987, 'Early several trends in plant communities on a surface coal mine in southwestern Wyoming', *Vegetation*, vol. 73, pp. 21 - 29.

Hieberler, D. 1994, 'The Swarm simulation system and individual - based modeling', in proceeding of *Decision Support 2001: Advanced Technology for Natural Resource Management*, Toronto.

Hobbs, R.J. 1999, 'Restoration of disturbed ecosystems', in *Ecosystems of the world; 16 Ecosystems of disturbed ground*, eds L.R. Walker, pp. 673 - 683.

Hobbs, R.J. & Harris, J.A. 2001, 'Restoration ecology: Repairing the earth's ecosystems in the new millennium', *Restoration Ecology*, vol. 9, pp. 239 - 246.

Hobbs, R.J. & Norton, D.A. 1996, 'Towards a conceptual framework for restoration ecology', *Restoration Ecology*, vol. 4, pp. 93 - 110.

Hodgekinson, K.C. & Oxley, R.E. 1990, 'Influence of fire and edaphic factors on germination of the arid zone shrubs *Acacia Aneur*, *Cassia nemophila* and *Dodonaea viscosa', Australian Journal of Botany*, vol. 38, pp. 269 - 279.

Hodgekinson, K.C., Harrington, G.N. & Miles, G.E. 1980, 'Composition, spatial and temporal variability of the soil seed pool in a *Eucalyptus populnea* shrub woodland in central New South Wales', *Aust. J. Ecol.*, vol. 5, pp. 23 - 29.

Hodgkinson, K.C. & Griffin, G.F. 1982, 'Adaptation of shrub species to fire in the arid zone', in *Evolution of the Flora and Fauna of Arid Australia,* eds W.R.Barker &. P.J.M.Greenslade, Peacock publications: South Australia.

Holling, C.S. 1973, 'Resilience and stability of ecological systems', *Annu. Rev. Ecol. Syst.*, vol. 4, pp. 1 - 23.

Holling, C.S. 1996, 'Engineering resilience versus ecological resilience', in *Engineering Within Ecological Constraints,* eds. P. Schulze, National Academy Press, Washington, DC, pp. 31 - 43.

Hore-Lacy, I. (eds) 1979, *Mining Rehabilitation*, Australian Mining Industry Council, Canberra.

Horn, H.S. 1975, 'Markovian processes of forest succession', in *Ecology and Evolutions of Communities,* eds M.L. Cody & J.M. Diamond, Massachusetts, Cambridge, pp. 196 - 211.

Howard, P.J. & Howard, D.M. 1990. 'Use of organic carbon and loss-on-ignition to estimate soil organic matter in different soil types and horizons', *Biology and Fertility of Soils*, vol. 9, pp. 306 - 310.

Huston, M. 1979, 'A general hypothesis of species diversity', *American Nature*, vol. 113, pp. 81 - 103.

Jackson, L.J. 1991, *Surface coal mines - restoration and rehabilitation,* IEA Coal Research, London.

Jacobs, M.R. 1955, *Growth Habits of the Eucalypts,* Commonwealth Government Printer, Canberra.

Jenkins, D., Brescacin, C.R., Duxbury, C.V., Elliott, J.A., Evans, J.A., Grablow, K.R., Hillegass, M. & Lyon, N.B. *et al.*, 2007. ' Does size matter for dispersal distance?', *Global Ecology and Biogeography*, vol. 16, pp. 415 - 425.

Johnson, K.H., Vogt, K.A., Clark, HJ., Schmitz, O.J. & Vogt, DJ. 1996, 'Biodiversity and the productivity and stability of ecosystems', *Trends Ecol. Evol.*, vol. 11, pp. 372 - 377.

Johnson, M.S., Cooke, J.A. & Stevenson, J.K. 1994, 'Revegetation of metalliferous wastes and land after metal mining', in *Mining and its Environmental Impact,* eds R.E. Hester & R.M.Harrison, Vol. Issues in Environmental Science and Technology, Royal Society of Chemistry, Letchworth, England, pp. 31 - 48.

Jordan, W.R., Gilpin, M.E. & Aber, J.D. 1987, *Restoration Ecology: A synthetic Approach to Ecological Research,* Cambridge University Press.

Krebs, C.J. 1972, *Ecology: The Experimental Analysis of Distribution and Abundance* Harper Internal Edition, New York.

Koch, J.M. 1987, 'Nitrogen accumulation in a rehabilitated bauxite-mined area in the Darling Range, Western Australia', *Australian Forest Research* vol. 17:11, pp. 59 - 72.

Koch, J.M. & Ward, S.C.. 1994, 'Establishment of understorey vegetation for rehabilitation of bauxite-mined areas in the jarrah forest of Western Australia', *Journal of Environmental Management*, vol. 41, pp. 1 - 15.

Kohler, P. & Huth, A. 1998, 'The effects of tree species grouping in tropical rainforest modeling-simulations with the individual-based model FORMIND', *Eco. Modelling*, vol. 109, pp. 301 - 321.

Lacey, C.J. 1973, 'Silvicultural characteristics of white cypress pine', *Forestry Commission of New South Wales, Research Note*, 26.

Law, R. & Dieckmann, U. 2000. 'A dynamical system for neighborhoods in plant communities', *Ecology*, vol. 81, pp. 2137 - 2148.

Larson, JA. & Franklin, FJ. 2006. 'Structural segregation and scales of spatial dependency in *Abies amabilis* forests', *Journal of Vegetation Science*, 17(4), pp. 489 - 498.

Lhotka, L., 1994, 'Implementation of individual-oriented models in aquatic ecology', *Eco. Model.*, vol. 74, pp.47 - 62.

Linkdenmayer, D.B. & Franklin, J.F., 2002, *Conserving Forest Biodiversity,* Island Press, Washington, DC.

Little LR, 2002, Investigating competitive interactions from spatial patterns of trees in multispecies boreal forests: the random mortality hypothesis revisited. *Canadian Journal of Botany,* 80 (1): 93-100

Loch, R.J. & Orange, D.N., 1997, 'Changes in some properties of topsoil at Tarong Coal-Meandu Mine coalmine with time since rehabilitation', *Australian Journal of Soil Research,* vol. 35, pp.777 - 784

Lömnicki, A. 1999, 'Individual-based models and the individual-based approach to population ecology', *Ecological Modelling*, vol. 115, pp. 191 - 198.

Lorek, H. & Sonnenschein M. 1998, 'Object-oriented support for modeling and simulation of individual-oriented ecological models', *Ecological Modelling*, vol. 108, pp. 77 - 96.

Lotka, A.J. 1925, *Elements of physical biology,* Baltimore: Williams & Wilkins Co.

Luken, J.O. 1990, *Directing Ecological Succession,* Chapman & Hall, London.

Luzuriaga, A.L., Escudero, A., Olana,J.M., & Loidi, J. 2004, 'Regenerative role of seed banks following an intense soil disturbance', *Acta Oecologica-International Journal of Ecology*, vol. 27, pp.57 - 66.

MacArthur, R. H. and Wilson, E. O. 1967. *The Theory of Island Biogeography*. Princeton, N.J.: Princeton University Press.

MacArthur, R. H. & Wilson, E. O. 2001, *The theory of Island Biogeography, with a new preface by Edward O. Wilson*, Princeton University Press.

MacArthur, R.H., 1972. *Geographical Ecology: Patterns in the Distribution of Species*, Harper & Row, New York.

Mandel, J. 1964, *The Statistical Analysis of Experimental Data*, Interscience, London.

Margalef, R. 1958, 'Information theory in ecology', *Gen. Syst.*, vol. 3, pp. 36 - 71.

Margalef, R. 1963, 'On certain unifying principles in ecology', *American Nature*, vol. 97, pp. 357 - 374.

Margalef, R. 1968, *Perspecitives in Ecological Theory,* University of Chicago Press, Chicago.

Martinez-Ruiz, C., Fernandez-Santos, B., Putwain, PD. & Fernandez-Gomez, MJ. 2007. 'Natural and man-induced revegetation on mining wastes: Changes in the floristic composition during early succession', *Ecological Engineering*, 30(3), pp. 286 - 294.

Martineau, Y. & Saugier, B, 2007, ' A process-based model of old field succession linking ecosystem and community ecology', *Ecological modelling*, 204(3-4), pp. 399 - 419.

Mineral Council of Australia (eds), 2002, 'Sustainable Development Report', Mineral Council of Australia.

McCook, L.J. 1994, 'Understanding ecological community succession: causal models and theories, a review', *Vegetation*, vol. 110, pp. 115 - 147.

McIntosh, R.P. 1981, 'Succession and ecological theory', in *Forest Succession: Concept and Application*, eds D.C. West, H.H. Shugart & D.B. Botkin, Springer-Verlag New York, Inc., New York.

McNamara, R., Lefebvre, N. & Joyce, J., 1999, 'Assessment of mine-site rehabilitation performance at the Oaky Creek coal mine, Bowen Basin, central Queensland', in (eds C.J., Asher & L.C. Bell), *Proceedings of the Workshop on Indicators of Ecosystem Rehabilitation Success*, Melbourne, 23 - 24 October 1998, Australian Centre for Mining Environmental Research, Brisbane, pp. 125 - 137

Miles, J. 1987, 'Vegetation succession: Past and present perceptions', in *Colonization, Succession and Stability,* eds A. J. Gray, M.J. Crawley, M.J. & Edwards, P.J., Blackwell, Oxford, pp. 1 - 29.

Mooij, W.M. & Boersma, M. 1997, 'An object-oriented simulation framework for individual-based simulation (OSIRIS): daphnia population dynamics as an example', *Ecological Modelling*, vol. 93, pp. 139 - 153.

Moravec. 1990, 'Regeneration of N.W. Africa *Pinus halepensis* forests following fire', *Vegetation*, vol. 87, pp. 29 - 36.

Morin, K.A., & Hutt, N.M. 1999, 'Prediction of drainage chemistry in post-mining landscapes using operational monitoring data', *Ecology of Post-Mining Landscape,* 15 - 19 March 1999, Cottbus, Germany.

New, T. R. 1984, A Biology of Acacias, Oxford University Press

Norman, M.A. & Koch, J.M. 2005, 'Different in species abundance between sites rehabilitation with direct and stockpiled soil', Environmental Department Research Bulletin, Alcoa World Alumina Australia.

Noble, I.R. & Slatyer, R.O. 1981, 'Concepts and models of succession in vascular plant communities subject to recurrent fire', in *Fire and Australia biota*, eds A.M. Gill, R.H.Groves & I.R.Noble, Australian National Committee for SCOPE, Australian Academy of Science.

Odum, E.P. 1969, 'The strategy of ecosystem development', *Science*, vol. 164, pp. 262 - 70.

O'Reilly, T., 1994, Outlook for Australian coal in the Asia Pacific, *Outlook 94*, ABARE, Australia

Osawa, A. 1992, 'Development of a mixed-conifer forest in Hokkaiko, northern Japan following a catastrophic wind storm: A 'parallel' model of plant succession', in *The ecology and silviculture of mixed-species forests.* Kluwer Academy Publishers.

Pacala, S.W. & Deutschman, D.H. 1995. Details that matter: The spatial distribution of individual trees maintains forest ecosystem function. *Oikos*, vol. 74, pp. 357 - 365.

Pacala, S.W., Canham, C.D. & Silander, J.A.J. 1993. Forest models defined by field measurements: I. The design of a northeastern forest simulator. *Can. J. For. Res,* vol. 23, pp. 1980 - 1988.

Pacala, S.W., Canham, C.D., Silander, J.A.J., Kobe, R.K. & Ribbens, E. 1996. Forest models defined by field measurements: Estimation, error analysis and dynamics. *Ecological Monographs*, vol. 66, pp. 1 - 43.

Parker, M.T. 1998. *Ascape*. Retrieved 20th Nov. 2006 from http://www.brook.edu/es/dynamics/models/ascape. The Brookings Institution, Washington DC.

Parker, M.T. 2001, 'What is Ascape and why should you care?', *Journal of Artificial Societies and Social Simulation*, vol. 4, no. 1, retrieved from http://www.soc.surrey.ac.uk/JASSS/4/1/5.html

Penfold, A.R. & Willis, J.L. 1961, *The Eucalypts: Botany, Cultivation, Chemistry, and Utilization*, Interscience publishers, INC., New York.

Pickett, S.T.A. & White, P.S.(eds) 1985, *The ecology of natural disturbance and patch dynamics*, Academy Press, Orlando.

Pratt, J.R. & Stevens, J. 1992, 'Restoration Ecology: repaying the national debt', in High Altitude Revegetation Workshop, eds W.G. Hassell, S.K. Nordstrom, W.R. Keammerer & W.J. Todd, Colorada State University, Vol. No. 10, pp. 40 - 49

Pronk TE, During HJ & Schieving F, 2007, Coexistence by temporal partitioning of the available light in plants with different height and leaf investments, *Ecological Modelling,* 204 (3-4): 349-358

Purdie, R.W.& Slatyer.R.O. 1976, 'Vegetation succession after fire in sclerophyll woodland communities in south-eastern Australia', *Aust. J. Ecol.*, vol. 1, pp. 223 - 236.

Randell, B. R. 1970, 'Adaptations in the genetic system of Australian arid-zone Cassia species', *Aust. J. Bot. vol.* 18, pp. 77 - 97.

Rhoades, J.D., N.A. Manteghi, P.J. Shouse, and W.J. Alves. 1989. 'Soil electrical conductivity and soil salinity: new formulations and calibrations'. *Soil Sci. Soc. Am J.* vol. 53, pp. 433 - 439

Richards, F.J. 1959, 'A flexible growth function for empirical use', *Journal of Experimental Biology*, vol. 1, pp. 290 - 300

Rodrigues, R.R., Marins, S.V. & Barros, L.C. 2004. 'Tropical rain forest regeneration in an area degraded by mining in Mato Grosso State, Brazil', *Forest Ecology and Management*, vol. 190, pp. 323 - 333.

Sánchez- Belásquez, L. R. 2003. A model to infer succession mechanisms in forests. *Agrociencia*. vol. 37 (5). pp. 533 - 543.

Savage, M. & Askenazi, M. 1998, 'Arborscapes: A swarm-based multi-agent ecological disturbance model', *Geographical and Environmental Modeling*. submitted, available as Santa Fe Institute Working Paper 98-06-056.

Savage, M., Sawhill, B. & Askenazi, M. 2000, 'Community dynamics: What happens when we rerun the tape?', *Journal of Theory Biology,* vol. 205, pp. 515 - 526

Schmid, M. 2006. 'Domination and colonisation of nature: an approach of environmental history to integrate the material and the symbolic', *Mitteilungen der osterreichischen geograhischen gesellschaft.* vol.148. pp. 57 - 74.

Schmitz, O.J. 2000, 'Combining field experiments and individual-based modeling to identify the dynamically relevant organizational scale in a field system', *Oikos*, vol. 89, pp. 471 - 484.

Schneider, D. 2001. ' The rise of the concept of scale in ecology'. *Bioscience*, vol. 51, No. 7, pp. 545 - 553.

Schnitzler, A. & Closset, D. 2003. Forest dynamics in unexploited birch (*Betula pendula)* stands in the Vosges (France): structure, architecture and light patterns. *Forest Ecology and Management*. vol. 183. pp. 205 - 220.

Schwartz, M.W., Hermann, S.M. & Van Mantgem, P.J. 2000, 'Population persistence in Florida torreya: Comparing modeled projections of a declining coniferous tree', *Conservation Biology*, vol. 14, pp. 1023 - 1033.

Scullion, J. 1992, 'Re-establishing life in restored topsoils', *Land Degradation and Rehabilitation*, vol. 3, pp. 161 - 168.

SER, 2002, Society for Ecological Restoration (SER) International Science & Policy Working Group (eds.), *The SER primer on ecological restoration.* http://www.ser.org

Sharov, A. 1996, *Quantitative population ecology (online lecture)* http://www.ento.vt.edu/~sharov/PopEcol/popecol.html, Department of Entomology, Virginia Tech, Blacksburg, VA.

Shugart, H.H. 1984, *A Theory of Forest Dynamics: The Ecological Implications of Forest Succession Model*, Springer, New York, pp. 278.

Shugart, H.H. & West, D.C. 1977, 'Development of an Appalachian deciduous forest succession model and its application to assessment of the impact of the chestnut blight', *Journal of Environmental Management*, vol. 5, pp. 161 - 170.

Smith, A. & Barton, B. 1991, 'Some consideration in the prediction and control of acid mine drainage impact on groundwater from mining in North America', in conference proceedings of EPPIC Water Symposium conference, Johannesburg, pp. 16 - 17

Snäll, T., Ribeiro, P.J. & Rydin, H. 2003. 'Spatial occurrence and colonisations in patch - tracking metapopulations: local conditions versus dispersal', *OIKOS*, vol. 103, pp. 566 - 578.

Statsoft, 2001, 'Statistica', Last retrieved 20th Nov. 2005 from http://www.statsoft.com

Tabor, J., McElhinny, C., Hickey, J. & Wood, J. 2007. 'Colonisation of clearfelled coupes by rainforest tree species from mature mixed forest edges, Tasmania, Australia', *Forest Ecology and Management.* vol. 240. pp. 13 - 23.

Tansley, A.G. 1935, 'The use and abuse of vegetational concepts and terms', *Ecology*, vol. 16, pp. 284 - 307.

Uchmanski & Grimm, V., 1996. Individual-based modelling in ecology: what makes the difference? TREE.

Urban, D.L., Bonan, G.B., Smith, T.M. & Shugart, H.H. 1991. 'Spatial applications of gap models', *Forest Ecology and Management,* vol. 42. pp 95 - 110.

Urban, D.L. & Shugart, H.H. 1992, 'Individual-based models of forest succession', in *Plant Succession: theory and prediction*, eds D.C. Glen-Lewin, R.K. Peet & T.T. Veblen, Chapman & Hall, London.

van Andel, J., Bakker, J.P.B.& Grootjans, A.P. 1993, 'Mechanisms of vegetation succession: a review of concepts and perspectives', *Acta Botanica Neerlandica*, vol. 42, pp. 413 - 433.

Vogel, R.M. 1999. 'Stochastic and deterministic world views', *Journal of Water Resource Planning and Management,* 125(6), pp. 311 - 313.

Vogt, K.A., 1997, *Ecosystems: Balancing Science with Management*, Springer, New York

Volterra, V. 1926. 'Variations and fluctuations of the numbers of individuals in animal species living together'. in *Animal Ecology,* edn, Chapman, R.N., pp. 409 - 448. McGraw Hill (Reprinted by McGraw Hill 1931), New York.

Walker, L.R. & Moral, R.D. 2003, *Primary Succession and Ecosystem Rehabilitation.* Cambridge University Press, Cambridge.

Ward, S.C., Koch, J.M. & Ainsworth, G.L. 1996, 'The effect of timing of rehabilitation procedures on the establishment of a jarrah forest after bauxite mining', *Restoration Ecology*, vol. 4, pp. 19 - 24.

Watt, A.S. 1947, 'Pattern and process in the plant community', *Journal of Ecology*, vol. 35, pp. 1 - 22.

Westoby, M., Walker, B.H. & Noy-Meir, I. 1989, 'Opportunistic management for rangelands not at equilibrium', *Journal of Range Management*, vol. 42, pp. 266 - 274.

Whitemore, T.C. 1989, 'Canopy gaps and two major groups of forest trees', *Ecology*, vol. 70, pp. 536 - 538.

Whittaker, R.H. 1953, 'A consideration of climax theory: the climax as a population and pattern', *Ecol. Monogr.*, vol. 23, pp. 41 - 78.

Wiegleb, G. & Felinks, B. 2001. ' Primary succession in post-mining landscapes of Lower Lusatia - chance or necessity', *Ecological Engineering,* vol. 17, pp. 199 - 217.

Wills, T. J. & Read, J. 2007. 'Soil seed bank dynamics in post-fire heathland succession in south-eastern Australia', *Plant Ecology*. vol. 190 (1). pp. 1 - 12.

Winkler, E. & Heinken, T. 2007. Spread of an ant-dispersed annual herb: An individual - based simulation study on population development of *Melampyrum pratense L. Ecological Modelling*. vol. 203, pp. 424 - 438.

Wiram, V.P. 1979, 'Soil-overburden Characterisation of Boggabri Coal Prospect Area, Boggabri, NSW, Australia', AMAX Coal Company, Indianapolis, Indiana.

Withers, J.R. 1978, 'Studies on the status of unburnt *Eucalptus* woodland at Ocean Grove, Victoria. II. The differential seedling establishment of *Eucalyptus ovata labill.* and *Casuarina littoralis Salisb*', *Aust. J. Bot.*, vol. 26, pp. 465 - 483.

Yates, C.J., & Hobbs, R. 1997, 'Woodland Restoration in the Western Australian Wheatbelt: A Conceptual Framework Using a State and Transition Model', *Restoration Ecology*, vol. 5, pp. 28 - 35.

E. pilligensis, E. albens on both northern tospoiled area (right) and the surrounding natural forest (left)

Northern bare overburden area in 1981(Source: J.A.Duggin).

Regeneration on northern bare overburden, 2002

Callitris glaucaphylla growth on southern bare overburden, 2002

Shrub colonisation on topsoiled area 1981 (Source from J.A.Duggin).

Regeneration on northern topsoiled area: *E. crebra* (the tallest tree in the middle) with some small *C. glaucophylla*, *S. nemophila*, and *E. crebra* at Boggabri, 2002

Callitris glaucophylla saplings on northern topsoiled area, 2002

Flow chart for programming

```
/* To record all of the attributes of a species and to execute the actural
 * simulation step on behalf of every tree belonging to the species.
 * The attributes are included as:
 * Age level, Canopy structure, resitance of tolerance, seedPeriod, Seeding radius,
 * and seedMortalityRate etc.
*/

 import java.awt.Color;
 import uchicago.src.sim.space.Object2DGrid;
 import uchicago.src.sim.space.RasterSpace;

 public class Species{
 private RasterSpace space;
 private Object2DGrid seedGrid;

 private Plant tree;
 String  speciesName;
 private Color speciesColor;
 private double x, y;
 private int initialPopulation;
 private int maxAge;
 private int seedingPeriod;
 private int seedingRadius;
 private int seedsPerCell;
 private double mort1, mort2, mort5, mort10;
 private double heightMax;
 private double growRate;
 private int numberOfTrees;
 private int totalTrees;
 int ageLevel[] = new int [3];

 public Species(String speciesN, Color spColor, Object2DGrid seedGrid,

            RasterSpace space) {
  this.space = space;
  this.seedGrid = seedGrid;
  speciesName = speciesN;
  this.seedGrid = seedGrid;
  speciesColor =spColor;

 }

 public void setXY(double x, double y) {
   this.x = x;
   this.y = y;
 }


 public void setX(double x) {
   this.x = x;
 }

 public void setY(double y) {
   this.y = y;
 }

 public int getX() {  return space.getCellCol(x); }
 public int getY() {  return space.getCellRow(y); }
 public void setMaxAge(int age) { maxAge = age;  }
 public int getMaxAge()  { return maxAge;  }
```

```
public void setAgeLevel(int ageLevel[]) { this.ageLevel = ageLevel; }
public int[] getAgeLevel() { return ageLevel;  }
public void showAgeLevel() {
  System.out.println(ageLevel[0]+" "+ageLevel[1]+" "+ageLevel[2]+" "+ageLevel[3]);
}

public void setHeightMax(double htmax) { heightMax = htmax; }
public double getHeightMax() { return heightMax; }
public void setGrowRate(double gr){ growRate= gr;  }
public double getGrowRate() { return growRate; }
public void setMort1(double mt1){ mort1 = mt1;  }
public double getMort1(){  return mort1;  }
public void setMort2(double mt2){ mort2 = mt2;  }
public double getMort2(){  return mort2;   }
public void setMort5(double mt5){ mort5 = mt5;  }
public double getMort5(){  return mort5;  }
public void setMort10(double mt10){ mort10 = mt10;  }
public double getMort10(){ return mort10; }
public void setSeedingPeriod(int seedingPeriod) {
  this.seedingPeriod = seedingPeriod;
}

public int getSeedingPeriod() {
  return seedingPeriod;
}

public void setSeedingRadius(int seedingRadius) {
  this.seedingRadius = seedingRadius;
}

public int getSeedingRadius() {
  return seedingRadius;
}


public void setSeedsPerCell(int spc){ seedsPerCell = spc;  }
public int getSeedsPerCell(){ return seedsPerCell;  }
public void setSpeciesName(String sn){ speciesName = sn;  }
public String getSpeciesName(){ return speciesName ;  }
public void setColor(Color c){ speciesColor = c;  }
public Color getColor(){  return speciesColor;  }

public int getInitialPopulation() {
  return initialPopulation ;
}

public Object2DGrid getSeedGrid(){
  return seedGrid ;
}

public void setTotalTrees(int aTotal) {
  totalTrees = aTotal ;
}

public int getTotalTrees() {
  return totalTrees;
}

public int getLevelAtAge(int theAge) {
  int i ;
```

```
    for(i = 0 ; i < 4 ; i++)
      if(theAge <= ageLevel[i]) break ;
          return i ;
  }

  public void incrementPopulation(){
    numberOfTrees++ ;
  }

  // ** for the future analysis
  public double getRelativeProportion() {
     return ((double) numberOfTrees) / ((double) totalTrees) ;
  }

/*

  public void seeding(Tree t) {
   int i,j, x, y;
   int seedingCounter ;
   x = t.getX() ;
   y = t.getY() ;
   seedingCounter = t.getSeedingCounter() ;
   if(seedingCounter == seedingPeriod){
     for(i = x - seedingRadius; i <= x + seedingRadius; i++)
       for(j = y - seedingRadius; j <= y + seedingRadius ; j++)
         if( (i >= 0) && (j >= 0) && (i < seedGrid.getXSize()) && (j < seedGrid.getYSize()))
           seedGrid.addSeedsAt(i, j, seedsPerCell);
   }
   seedingCounter++;
 // aTree.setSeedingCounter(0);
  }
*/


public int spNameSeries(){

   int k;
   k = 0;
   if(speciesName == "E. pilligaensis" )    { k = 0;};
   if(speciesName == "E. crebra" )          { k = 1;};
   if(speciesName == "E. alben" )           { k = 2;};
   if(speciesName == "Callitris glaucophylia") { k = 3;};
   if(speciesName == "Casuarina cristata") { k = 4;};
   if(speciesName == "E. populnea")         { k = 5;};
   if(speciesName == "Acacia deanei" )      { k = 6;};
   if(speciesName == "Cassia nemophila")    { k = 7;};
   if(speciesName == "Dodonaea viscosa")    { k = 8;};
   return k;
}

 /* public void destroySeedsAt(int theX, int theY) {

    seedGrid.putValueAt(theX, theY, 0) ;

  }*/
}
```

```
// **the record of the state of an individual tree, including its life history
// attributes.
import java.awt.Color;
import uchicago.src.sim.gui.Drawable;
import uchicago.src.sim.gui.SimGraphics;
import uchicago.src.sim.space.*;
//public class Tree extends Species implements Drawable{
public class Plant implements Drawable {
   private SeedSpace space;
   private double x, y;
   private double xOrig;
   private double yOrig;
   private double xTerm;
   private double yTerm;
   private int col, row;
   private int treeAge;
   private int maxAge;
   private int seedingPeriod;
   private int seedsPerCell, seedingRadius;
   private double growRate;
   private double height, heightMax;
   private String speciesName;

   private Color myColor;

   int ageLevel[] = new int[3];
   public Plant(double x, double y, int age, String speciesName,
                    Color mycolor, RasterSpace space) {
      this.x = x;
      this.y = y;
      this.space = space;
      this.col = space.getCellCol(x);
      this.row = space.getCellRow(y);
      xOrig = space.getOriginX();
      yOrig = space.getOriginY();
      xTerm = space.getTermX();
      yTerm = space.getTermY();
      this.treeAge = age;
      this.speciesName = speciesName;
      this.myColor = mycolor;
   }

   public void setX(double x) {
      this.x = x;
   }

   public void setY(double y) {
      this.y = y;
   }

   public int getX() {
      return space.getCellCol(x);
   }

   public int getY() {
      return space.getCellRow(y);
   }

   public void setXI(int col){
         this.col = col;
```

```
}
    public void setYI(int row){
   this.row = row;
}

public int getXI(){
   return this.col;
}

public int getYI(){
   return row;
}

public void setColor(Color c) {
   myColor = c;
}

public Color getColor() {
   return myColor;
}

public void setSpeciesName(String sn) {
   this.speciesName = sn;
}

public String getSpeciesName() {
   return speciesName;
}

public void setTreeAge(int atreeAge) {
   treeAge = atreeAge;
}

public int getTreeAge() {
   return treeAge;
}

public void setMaxAge(int age) {
   maxAge = age;
}

public int getMaxAge() {
   return maxAge;
}

public void setHeight(double ht) {
   height = ht;
}

public double getHeight() {
   return height;
}

public void setHeightMax(double htm) {
   heightMax = htm;
}

public double getHeightMax() {
   return heightMax;
}
```

```
/*
 * public void setStemDiameter(double diameter){ stemDiameter = diameter; }
 *
 * public double getStemDiameter(){ return stemDiameter; }
 *
 */

public void setGrowRate(double gr) {
   growRate = gr;
}

public double getGrowRate() {
   return growRate;
}

public void setSeedingPeriod(int seedingPeriod) {
   this.seedingPeriod = seedingPeriod;
}

public int getSeedingPeriod() {
   return seedingPeriod;
}

public void setSeedingRadius(int seedingRadius) {
   this.seedingRadius = seedingRadius;
}

public int getSeedingRadius() {
   return seedingRadius;
}

public void setSeedsPerCell(int seedsPerCell) {
   this.seedsPerCell = seedsPerCell;
}

public int getSeedsPerCell() {
   return seedsPerCell;
}

public int incrtreeAge() {
   return treeAge++;
}

public void setAgeLevel(int ageLevel[]) {
   this.ageLevel = ageLevel;
}

public int[] getAgeLevel() {
   return ageLevel;
}

public void showAgeLevel() {
   System.out.println(ageLevel[0] + " " + ageLevel[1] +
   " " + ageLevel[2]);
}

public void calculator(String name, double a, double b, double c)
{
   double h;// = this.getHeight();
```

```
        h = a * (Math.pow(1 - Math.exp(b * this.treeAge), c));
        //h = (int)(h*10)/10;
        h = (double)((int)(h*100))/100;

        this.setHeight(h);
       // System.out.println(name + ":\t" + this.treeAge + " age ---- height  " + height);
    }

    public void step()
    {
        String name = (String) this.getSpeciesName();
        if (height < this.getHeightMax()) {
            if (name == "E. pilligaensis") {
                this.calculator(name, 22.10537, -0.11555, 2.230091);
            }
            if (name == "E. crebra") {
                this.calculator(name,30.09728, -0.041867, 1.481939);
            }
            if (name == "E. albens") {
                this.calculator(name,26.43896, -0.094419, 2.998853);
            }
            if (name == "E. populnea") {
                this.calculator(name,23.01912, -0.05, 1.398892);
            }
            if (name == "Casuarina cristata") {
                this.calculator(name,19.02120, -0.046469, 1.677284);
            }
            if (name == "Callitris glaucophylla") {
                this.calculator(name, 23.15374,-0.01559, 1.119820);
            }
            else if (name == "Acacia deanei" ||name == "Senna nemophila"|| name == "Dodonaea viscosa"){
                height += (double) this.getGrowRate();
                height = (double)((int)(height*100))/100;
            }
        }

        else {
            height = this.getHeightMax();
        }

        treeAge++;

    }

    public void draw(SimGraphics g) {
        // g.drawFastRect(super.getColor());
        g.drawFastRect(this.getColor());
    }
}
```

```
/* This model is developed under RePast (University of Chicago).
 * It simulates the forest succession after open-cut mining. The study site
 * locates in Boggabri, NSW
 * Any questions to this model, please contact Xianfeng Su
 */

import java.util.ArrayList;
import java.awt.Color;
import java.util.Vector;
import java.io.*;

import uchicago.src.sim.engine.*;
import uchicago.src.sim.gui.*;
import uchicago.src.sim.util.Random;
import uchicago.src.sim.space.*;
import uchicago.src.sim.analysis.*;

public class ForestModel extends SimModelImpl{

 //  creat arraylist to store seeds, mature species, and seedlings

 private ArrayList saplingList1 = new ArrayList();
 private ArrayList saplingList2 = new ArrayList();
 private ArrayList maturePlantList = new ArrayList();
 private ArrayList dominatedList = new ArrayList();
 private ArrayList crownList = new ArrayList();
 private ArrayList tL = new ArrayList();
 private ArrayList shrubL = new ArrayList();

 //to creat different layers
 private ArrayList firsdominatedList = new ArrayList();
 private ArrayList secondList = new ArrayList();
 private ArrayList thirdList = new ArrayList();
 private ArrayList fourthList = new ArrayList();
 private ArrayList fifthList = new ArrayList();

 //  creat lattice as species living environment
 private RasterSpace space;
 private RasterSpace zones;
 private RasterSpace spaceCyp;
 private Object2DGrid treeGrid;   // to grow trees and shrubs, which dominate the cell.
 private Object2DGrid sapling1; // to grow saplings of trees & shrubs.
 private Object2DGrid sapling2; // to grow saplings of trees & shrubs.
 private Object2DGrid tGrid;      // to save dominate plants
 private Object2DGrid crownGrid;

 private SeedSpace seedGrid;   // be regarded as Seed grid to grow Seeds.
 private Object2DGrid tG; // to save only trees

 private Object2DGrid shrubG; // to save only shrubs

 //to save different layers
 private Object2DGrid firstG;
 private Object2DGrid secondG;
 private Object2DGrid thirdG;
 private Object2DGrid fourthG;
 private Object2DGrid fifthG;

 //  use vectors to store distance from edge
```

```
 // private Vector distance1, distance2, distance3, distance1c, distance2c;  //10,20,and  // 30m buffer zones, use
vectors to store different soil types
 private Vector topsoil, tsv,overburden;

 private Forest forest;
 private Plant plant;
 private Species species;

 // initial starting parameters of the model
 private int maxSeeds = 500;
 private int speciesNumber = 9;
 private int lifespan;

 private String speciesName[ ] = {"E. pilligaensis", "E. crebra", "E. albens",
                                  "E. populnea", "Casuarina cristata", "Callitris glaucophylla",
                                  "Acacia deanei", "Senna nemophila", "Dodonaea viscosa"};
 private Color speciesColor[ ] = {Color.blue, Color.black, Color.pink,
                                  Color.magenta, Color.gray, Color.red, Color.yellow,
                                  Color.orange, Color.green};

 private int numSeedsAt[ ];     // total numbers of seeds for all species at one cell.
 private int numSpecAt[ ][ ];    // total seeds for one certain species at a cell.

 private int seedSeriesAt[ ];

 int sp1, sp2, sp3, sp4, sp5, sp6, sp7, sp8, sp9, sp10, sp11;   // for population graph

  // A dataRecorder is used to record the data
  // generated by the simulation into a file
 private DataRecorder recorder1;
 private DataRecorder recorder2, recorder3, recorder4, recorder5, recorder6, recorder7, recorder8, recorder9,
  recorder10, recorder11;

 private Schedule schedule;
 private DisplaySurface dsurf; // for yTree1

 private DisplaySurface tsf; // display tree colonisation
 private DisplaySurface shrub; // dispplay shrub colonisation

 private DisplaySurface first;
 private DisplaySurface second;
 private DisplaySurface third;
 private DisplaySurface fourth;
 private DisplaySurface fifth;
 private DisplaySurface y2surf; //for saplingList2

 private OpenSequenceGraph graph; //a graph to show the change of species vs time
 //private OpenHistogram bar;

 public ArrayList getmaturePlantList() {
   return maturePlantList;
 }

 public ForestModel(){
 }

  // build the model based on the current parameters.
 private void buildModel() {
          try{
                    String file = "site.txt"; //
```

```
                    java.io.InputStream stream = new FileInputStream(file);
                    space = new RasterSpace(stream);
  }catch(Exception e){
    System.out.println(e);
  }  //read location &soil map into the model

  try{
   String file1 = "distance.txt";
   java.io.InputStream stream = new FileInputStream(file1);
   zones = new RasterSpace(stream);
  }catch(Exception e){
    System.out.println(e);
  } // read 10m buffer zone map into the model for spray seeds from Surounding forest

  try{
    String file2 = "distanceForCypress.txt";
    java.io.InputStream stream = new FileInputStream(file2);
    spaceCyp = new RasterSpace(stream);
   }catch(Exception e){
     System.out.println(e);
   }  //read community map into the model


  treeGrid = new Object2DGrid(space.getSizeX(), space.getSizeY()); //save tree trucks and shrubs
  sapling1 = new Object2DGrid(space.getSizeX(), space.getSizeY()); // saplings grows
  sapling2 = new Object2DGrid(space.getSizeX(), space.getSizeY()); // for sapling grows
  tGrid = new Object2DGrid(space.getSizeX(), space.getSizeY());  //tree truck and tree crown
  crownGrid = new Object2DGrid(space.getSizeX(), space.getSizeY()); // tree crown, tree truck and mature
shrubs
        tG = new Object2DGrid(space.getSizeX(), space.getSizeY());  // tree colonisation
        shrubG = new Object2DGrid(space.getSizeX(), space.getSizeY());  //shrub colonisation

        firstG = new Object2DGrid(space.getSizeX(), space.getSizeY());
        secondG = new Object2DGrid(space.getSizeX(), space.getSizeY());
        thirdG = new Object2DGrid(space.getSizeX(), space.getSizeY());
        fourthG = new Object2DGrid(space.getSizeX(), space.getSizeY());
        fifthG = new Object2DGrid(space.getSizeX(), space.getSizeY());

  //recorder = new DataRecorder( "./tree_data.txt", this);       // creat a new DataRecorder and

  //Record the 9 species data into files respectively     //Write the data to ./tree_data.txt
  recorder1 = new DataRecorder( "./pil.txt", this);
  recorder2 = new DataRecorder( "./cra.txt", this);
  recorder3 = new DataRecorder( "./alb.txt", this);
  recorder4 = new DataRecorder( "./pop.txt", this);
  recorder5 = new DataRecorder( "./cas.txt", this);
  recorder6 = new DataRecorder( "./cyp.txt", this);
  recorder7 = new DataRecorder( "./aca.txt", this);
  recorder8 = new DataRecorder( "./sen.txt", this);
  recorder9 = new DataRecorder( "./dod.txt", this);
  recorder10 = new DataRecorder( "./tt.txt", this);
  recorder11 = new DataRecorder( "./ts.txt", this);

  //  adds a data source that counts all the trees alive in the simulation.
  //  It does this by getting the size of the agendominatedList.

  class pilCount implements DataSource{
    public Object execute(){
      return new Integer(sp1);
    }
```

```
 }
 class Sp2Count implements DataSource{
   public Object execute(){
      return new Integer(sp2);
   }
 }
 class Sp3Count implements DataSource{
   public Object execute(){
      return new Integer(sp3);
   }
 }
 class Sp4Count implements DataSource{
   public Object execute(){
      return new Integer(sp4);
   }
 }
 class Sp5Count implements DataSource{
   public Object execute(){
      return new Integer(sp5);
   }
 }
 class Sp6Count implements DataSource{
   public Object execute(){
      return new Integer(sp6);
   }
 }
 class Sp7Count implements DataSource{
   public Object execute(){
      return new Integer(sp7);
   }
 }
 class Sp8Count implements DataSource{
   public Object execute(){
      return new Integer(sp8);
   }
 }
 class Sp9Count implements DataSource{
   public Object execute(){
      return new Integer(sp9);
   }
 }
 class Sp10Count implements DataSource{
   public Object execute(){
      return new Integer(sp10);
   }
 }
 class Sp11Count implements DataSource{
   public Object execute(){
      return new Integer(sp11);
   }
 }

recorder1.addObjectDataSource("pil", new pilCount());
recorder2.addObjectDataSource("sp2", new Sp2Count());
recorder3.addObjectDataSource("sp3", new Sp3Count());
recorder4.addObjectDataSource("sp4", new Sp4Count());
recorder5.addObjectDataSource("sp5", new Sp5Count());
recorder6.addObjectDataSource("sp6", new Sp6Count());
recorder7.addObjectDataSource("sp7", new Sp7Count());
recorder8.addObjectDataSource("sp8", new Sp8Count());
```

```
recorder9.addObjectDataSource("sp9", new Sp9Count());
recorder10.addObjectDataSource("sp10", new Sp9Count());
recorder11.addObjectDataSource("sp11", new Sp9Count());
// recorder.addObjectDataSource("Total Trees", new TreeCount());


//      INITIALIZATION OF FOREST MODEL

int k;
double x, y;
int xSize = space.getSizeX();
int ySize = space.getSizeY();

distance1 = new Vector();
distance2 = new Vector();
distance3 = new Vector();
distance1c = new Vector();
distance2c = new Vector();
topsoil = new Vector();
overburden = new Vector();
tsv = new Vector();
numSeedsAt = new int[xSize * ySize];
numSpecAt = new int[xSize * ySize][speciesNumber];
        seedSeriesAt = new int[xSize * ySize][speciesNumber];

// int chosenSeeds;


  // **initialize each cell value of latice as null
        seedGrid.initialise();

//    save all the cells from edge to central of 10, 20, 30m distance to 3 vectors seperately
for (x = zones.getOriginX(); x < zones.getTermX(); x = x + zones.getCellSize())
for (y = zones.getTermY(); y > zones.getOriginY(); y = y - zones.getCellSize()){
   if (zones.getValueAt(x, y) == 1 && space.getValueAt(x, y) != 5) {
            distance1.addElement(new Integer(space.getCellCol(x) + space.getCellRow(y) * xSize));
         }
   else if (zones.getValueAt(x, y) == 2 && space.getValueAt(x, y) != 5) {
                distance2.addElement(new Integer(space.getCellCol(x) + space.getCellRow(y) * xSize));
         }
      else if (zones.getValueAt(x, y) == 3 && space.getValueAt(x, y) != 5) {
                    distance3.addElement(new Integer(space.getCellCol(x) + space.getCellRow(y) * xSize));
            }
      else continue;
}

//    save all the cells of topsoil area into a vector
for (x = space.getOriginX(); x < space.getTermX(); x = x + space.getCellSize())
for (y = space.getTermY(); y > space.getOriginY(); y = y - space.getCellSize()){
         if (space.getValueAt(x, y) == 2) {
                    topsoil.addElement(new Integer(space.getCellCol(x) + space.getCellRow(y) * xSize));
         }
}

//     save all the cells of bare overburden into a vector.
for (x = space.getOriginX(); x < space.getTermX(); x = x + space.getCellSize())
for (y = space.getTermY(); y > space.getOriginY(); y = y - space.getCellSize()){
  if (space.getValueAt(x, y) == 3 ||space.getValueAt(x, y) == 4) { // for ob test
         overburden.addElement(new Integer(space.getCellCol(x) + space.getCellRow(y) * xSize));
  }
```

```
    }

  // spread seeds on topsoil.
          for ( k = 0; k < speciesNumber; k++){
            if((int)(topsoil*percent[k]) == 0) { continue; }
            else {
                    spreadSeeds(k, percent[k],topsoil, seedGrid);
            }
            clearGrid(seedGrid, topsoil);
          }


   //    spread seeds from Natrual forest into the study site ( 10m and 20m) -- no natual forest structure
influence
   //   first, spray 10m zone from edge to central

          for ( k = 0; k < SpeciesNumber; k++){
                    spread(distance1, seedGrid);
            spread(distance2, seedGrid);
            spread(distance3, seedGrid);
          }

          clearGrid(seedGrid);

   //   Last choose a seed at a site germinated randomly
   //   meanwhile these seeds chosen here are to grow up. clear all the objects on seedGrid.

   for (x = space.getOriginX(); x < space.getTermX(); x = x + space.getCellSize())
   for (y = space.getTermY(); y > space.getOriginY(); y = y - space.getCellSize()){
      randomSpring(x, y, sapling1, saplingList1);
      seedGrid.putObjectAt(space.getCellCol(x), space.getCellRow(y), null);
   }

 }


/*
 * This part is for the model display
 */

 public void buildDisplay() {

  ColorMap map = new ColorMap();
  for ( int i = 0; i < 9; i++) {

            map.mapColor ( i, i/2.15, 225, 250 );
  }

  Value2DDisplay rasterDisplay = new Value2DDisplay(space, map);
  rasterDisplay.setZeroTransparent(true);
  rasterDisplay.setDisplayMapping(1, 0);

  Object2DDisplay tDisplay = new Object2DDisplay(tGrid);
  tDisplay.setObjecdominatedList (dominatedList);
  dsurf.addDisplayable(rasterDisplay, " Gis Space" );
  dsurf.addDisplayableProbeable(tDisplay, "Vegetation Succession Model");


  Object2DDisplay tD = new Object2DDisplay(tG);
```

```
tD.setObjecdominatedList (tL); //display only trees
tsf.addDisplayable(rasterDisplay, " Gis Space" );
tsf.addDisplayableProbeable(tD, "Trees Colonisation");

Object2DDisplay shrubD = new Object2DDisplay(shrubG);
shrubD.setObjecdominatedList (shrubL); //display only trees
shrub.addDisplayable(rasterDisplay, " Gis Space" );
shrub.addDisplayableProbeable(shrubD, "Shrubs Colonisation");


Object2DDisplay firstDisplay = new Object2DDisplay(firstG);
firstDisplay.setObjecdominatedList (firsdominatedList);
first.addDisplayable(rasterDisplay, " Gis Space" );
first.addDisplayableProbeable(firstDisplay, "Height is higher than 20m");
//first.addDisplayableProbeable(firstDisplay, "Height over 15m");

Object2DDisplay secondDisplay = new Object2DDisplay(secondG);
secondDisplay.setObjecdominatedList (secondList);
second.addDisplayable(rasterDisplay, " Gis Space" );
second.addDisplayableProbeable(secondDisplay, "15-20m");
//second.addDisplayableProbeable(secondDisplay, "Height 2-15m");

Object2DDisplay thirdDisplay = new Object2DDisplay(thirdG);
thirdDisplay.setObjecdominatedList (thirdList);
third.addDisplayable(rasterDisplay, " Gis Space" );
third.addDisplayableProbeable(thirdDisplay, "10-15m");
//third.addDisplayableProbeable(thirdDisplay, "Height lower than 2m");

Object2DDisplay fourthDisplay = new Object2DDisplay(fourthG);
fourthDisplay.setObjecdominatedList (fourthList);
fourth.addDisplayable(rasterDisplay, " Gis Space" );
fourth.addDisplayableProbeable(fourthDisplay, "5-10m");

Object2DDisplay fifthDisplay = new Object2DDisplay(fifthG);
fifthDisplay.setObjecdominatedList (fifthList);
fifth.addDisplayable(rasterDisplay, " Gis Space" );
fifth.addDisplayableProbeable(fifthDisplay, "lower than 5m");

Object2DDisplay y2Display = new Object2DDisplay(sapling2);
y2Display.setObjecdominatedList (saplingList2);
y2surf.addDisplayable(rasterDisplay, " Gis Space" );
y2surf.addDisplayableProbeable(y2Display, "y 2");

addSimEvendominatedListener(dsurf);
// addSimEvendominatedListener(tsurf);


addSimEvendominatedListener(y2surf);

addSimEvendominatedListener(tsf); // tree colonisation
addSimEvendominatedListener(shrub); // shrub colonisation
addSimEvendominatedListener(first);
addSimEvendominatedListener(second);
addSimEvendominatedListener(third);
addSimEvendominatedListener(fourth);
addSimEvendominatedListener(fifth);

graph = new OpenSequenceGraph("Species Status", this);
graph.setXRange(0, 100);
graph.setYRange(0, 100);
```

```
            });
 graph.addSequence("Total Shrubs", new Sequence() {
              public double getSValue(){
                 return getSp11();
              }
            });
 graph.setAxisTitles("Year", "Numbers of Plants");

}


/*
 * * This is the main part. each step is one year. Firstly, spray from the natural forest into the
 * * study site, then check each cell. Each cell can only have one big occupied plant (Tree or shrub). If
 * The occupied by tree, under it, can have one sapling, under shrub no sapling.
 * *
 * */

public void step() {

 double x, y;
 int xSize = space.getSizeX();


//spray the first 10meter
 for ( k = 0; k < SpeciesNumber; k++){
                              spread(distance1, seedGrid);
                        spread(distance2, seedGrid);
                        spread(distance3, seedGrid);
          }
          clearGrid(seedGrid);

  //      clean seeds from seedSpace and keep it as empty!
  for (x = space.getOriginX(); x < space.getTermX(); x = x + space.getCellSize())
  for (y = space.getTermY(); y > space.getOriginY(); y = y - space.getCellSize()){
     seedGrid.putObjectAt(space.getCellCol(x), space.getCellRow(y), null);
  }
 /*
  * This part is recruitment part.  First, check the seeds, if older than 3 years, remove the seeds out
  * Second, germination Third, mortality
  */

 for (x = space.getOriginX(); x < space.getTermX(); x = x + space.getCellSize())
 for (y = space.getTermY(); y > space.getOriginY(); y = y - space.getCellSize()){

          int celln = space.getCellCol(x) + space.getCellRow(y) * xSize;  // get cell series number
          //move the seeds out from seed bank if it is older than 3 years
          for(int i = 0; i < speciesNumber; i++){
                    if(((numSpecAt[celln][i])/2 >= seedsPerCell[i])){
                                        numSpecAt[celln][i] = numSpecAt[celln][i] - seedsPerCell[i];
                                        numSeedsAt[celln] = numSeedsAt[celln] - seedsPerCell[i];
                    }
                    else continue ;
          }

          // germination
          if(crownGrid.getObjectAt(space.getCellCol(x), space.getCellRow(y)) == null ){
                              if(sapling1.getObjectAt(space.getCellCol(x), space.getCellRow(y)) == null ){
                                        randomSpring(x, y, sapling1, saplingList1);
```

```
                        }
                        else if (sapling2.getObjectAt(space.getCellCol(x), space.getCellRow(y)) == null ){
                                randomSpring(x, y, sapling2, saplingList2);
                        }

        }
        else{
                if(sapling1.getObjectAt(space.getCellCol(x), space.getCellRow(y)) == null){
                        Plant t = (Plant)crownGrid.getObjectAt(space.getCellCol(x), space.getCellRow(y));
                        if(spTeam(t) != 0 && sapling2.getObjectAt(space.getCellCol(x),
space.getCellRow(y)) == null ){
                                randomSpring(x, y, sapling1, saplingList1);
                        }
                }
        }


  }

  // check mortality of seedlings/saplings or seeds, for year 1, 2, 5,10 year
        mort(saplingList1, sapling1, mort1, 1);
        mort(saplingList2, sapling2, mort1, 1);
        mort(saplingList1, sapling1, mort2, 2);
        mort(saplingList2, sapling2, mort2, 2);
        mort(maturePlantList, treeGrid, mort2, 2);
        mort(saplingList1, sapling1, mort5, 3);
        mort(saplingList2, sapling2, mort5, 3);
        mort(maturePlantList, treeGrid, mort5, 3);
        mort(saplingList1, sapling1, mort10, 15);
        mort(maturePlantList, treeGrid, mort10, 15);
        mort(saplingList2, sapling2, mort10, 15);


        // Start checking the Grid, one cell after another.
  for (x = space.getOriginX(); x < space.getTermX(); x = x + space.getCellSize())
  for (y = space.getTermY(); y > space.getOriginY(); y = y - space.getCellSize()) {

   int col = space.getCellCol(x);
   int row = space.getCellRow(y);


   //See flowchart

   .......


  }

  plantGrowth();

  population();

   /*
   * This part for display the forest structure.
   */
 //save different height into different arraylist, so that they can be displayed in different layers
  firsdominatedList.clear();
  secondList.clear();
  thirdList.clear();
  fourthList.clear();
```

```
fifthList.clear();

for(double i= space.getOriginX(); i< space.getTermX(); i = i + space.getCellSize())
for(double j= space.getTermY(); j>space.getOriginY(); j= j-space.getCellSize()){
        if(crownGrid.getObjectAt(space.getCellCol(i), space.getCellRow(j)) != null){
                Plant t1 = (Plant)crownGrid.getObjectAt(space.getCellCol(i), space.getCellRow(j));
                double h1 = t1.getHeight();

                if(sapling1.getObjectAt(space.getCellCol(i), space.getCellRow(j)) != null){
                        Plant t2 = (Plant)sapling1.getObjectAt(space.getCellCol(i), space.getCellRow(j));
                        double h2 = t2.getHeight();

                        if (h1-h2 > 5){
                                if(h2 >= 15 && h2 <= 20){        secondList.add(t2);}

                                if(h2 >= 10 && h2 < 15){         thirdList.add(t2);   }
                                if(h2 >= 5 && h2 < 10){ fourthList.add(t2);          }
                                if(h2 < 5){        fifthList.add(t1);              }
                        }

                                if(h1 > 20){         firsdominatedList.add(t1);            }
                                if(h1 >= 15 && h1 <= 20){        secondList.add(t1);          }

                                if(h1 >= 10 && h1 < 15){         thirdList.add(t1); }
                                if(h1 >= 5 && h1 < 10){ fourthList.add(t1);          }
                                if(h1 < 5){        fifthList.add(t1);              }
                }

                else{
                        if(h1 > 20){        firsdominatedList.add(t1);            }
                        if( h1 >= 15 && h1 <= 20){secondList.add(t1);        }
                        if(h1 >= 10 && h1 < 15){ thirdList.add(t1); }
                        if(h1 >= 5 && h1 < 10){  fourthList.add(t1);          }
                        if(h1 < 5){                fifthList.add(t1);            }
                }

        } // there is a crown/tree in the cell, under it has/no saping

        else{ // nothing in crownGrid
                if(sapling1.getObjectAt(space.getCellCol(i), space.getCellRow(j)) != null){
                        Plant t1 = (Plant)sapling1.getObjectAt(space.getCellCol(i), space.getCellRow(j));
                        double h1 = t1.getHeight();
                        if (sapling2.getObjectAt(space.getCellCol(i), space.getCellRow(j)) != null){
                                Plant t2 = (Plant)sapling2.getObjectAt(space.getCellCol(i),
                                space.getCellRow(j));
                                  double h2 = t2.getHeight();
                                  if(h1 > h2 ){

                                        if(h1 >=10 && h1 < 15){
                                                thirdList.add(t1);
                                                if((h2 >= 5 && h2 < 10)){
                                                        fourthList.add(t2);
                                                }
                                                if(h2 < 5){
                                                        fifthList.add(t2);
                                                }

                                        }
                                        if(h1 >= 5 && h1 < 10){
                                                fourthList.add(t1);
```

```
                                        if(h2 < 5){
                                                fifthList.add(t2);
                                        }
                                }
                                if(h1 <5){
                                        fifthList.add(t1);
                                }
                        }
                        else{
                                if(h2 >= 10 && h2 < 15){
                                        thirdList.add(t2);
                                        if((h1 >= 5 && h1 < 10)){
                                                fourthList.add(t1);
                                        }
                                        if(h1 < 5){
                                                fifthList.add(t1);
                                        }

                                }
                                if(h2 >= 5 && h2 < 10){
                                        fourthList.add(t2);
                                        if(h1 < 5){
                                                fifthList.add(t1);
                                        }
                                }
                                if(h2 < 5){
                                        fifthList.add(t1);
                                }
                        }
                } // ytree2 != null and ytree1 != null

                else{ // one plant in sapling1
                        if(h1 >= 10 && h1 < 15){
                                thirdList.add(t1);
                        }
                        else if(h1 >= 5 && h1 < 10){
                                fourthList.add(t1);
                        }
                        if(h1 < 5){
                                fifthList.add(t1);
                        }
                }

            }
        }
}



//for display the dominate seedling/trees or one sapling in each cell
dominatedList.clear();
for(double i= space.getOriginX(); i< space.getTermX(); i = i + space.getCellSize())
for(double j= space.getTermY(); j>space.getOriginY(); j= j-space.getCellSize()){
        if(crownGrid.getObjectAt(space.getCellCol(i), space.getCellRow(j)) != null){
                Plant t = (Plant)crownGrid.getObjectAt(space.getCellCol(i), space.getCellRow(j));
                dominatedList.add(t);
        }
        else if(sapling1.getObjectAt(space.getCellCol(i), space.getCellRow(j)) != null){
                Plant t1 = (Plant)sapling1.getObjectAt(space.getCellCol(i), space.getCellRow(j));
```

```
                        dominatedList.add(t1);
                }
        }


        //  use tL to store all the trees and use tG to put in positions, and use tsf to display it
        //   to check maturePlantList and saplingList1 to find any tree in each potential cell
        tL.clear();
        for(int i = 0; i < dominatedList.size(); i++){
                Plant t = (Plant)dominatedList.get(i);
                int k = spTeam(t);
                if(k == 1){ tL.add(t);}
        }

        shrubL.clear();
        for(int i = 0; i < dominatedList.size(); i++){
                Plant t = (Plant)dominatedList.get(i);
                int k = spTeam(t);
                if(k == 0){ shrubL.add(t);}
        }


        //dsurf.updateDisplay();
        graph.step();
       // recorder.record();
        recorder1.record();
        recorder2.record();
        recorder3.record();
        recorder4.record();
        recorder5.record();
        recorder6.record();
        recorder7.record();
        recorder8.record();
        recorder9.record();
        recorder10.record();
        recorder11.record();

        dsurf.updateDisplay(); // forest succession
        y2surf.updateDisplay(); // second sapling layer
       // tsurf.updateDisplay();

        tsf.updateDisplay(); // tree colonisation
        shrub.updateDisplay(); // shrub colonisation
        first.updateDisplay();
        second.updateDisplay();
        third.updateDisplay();
        fourth.updateDisplay();
        fifth.updateDisplay();

       }

       public void plantGrowth(){
                        //     each plant will grow up by step() method
                for(int i = 0; i < saplingList1.size(); i++){
                                Plant t = (Plant)saplingList1.get(i);
                                if(t.getTreeAge() < t.getMaxAge()){
                                        t.step();
                                }
                                else{
                                                //saplingList1.remove(t);
```

```
                                        sapling1.putObjectAt(t.getX(), t.getY(), null);
                                        saplingList1.remove(i);
                                        saplingList1.trimToSize();

                        }
            }


            for(int i = 0; i < saplingList2.size(); i++){
                    Plant t = (Plant)saplingList2.get(i);
                        if(t.getTreeAge() < t.getMaxAge())  {
                                t.step();
                        }
                        else{
                                sapling2.putObjectAt(t.getX(), t.getY(), null);
                                //saplingList2.remove(t);
                                saplingList2.remove(i);
                                saplingList2.trimToSize();

                                }
            }


           for(int i = 0; i < maturePlantList.size(); i++){
                    Plant t = (Plant)maturePlantList.get(i);
                    if(t.getTreeAge() < t.getMaxAge()){ t.step();}
                    else {
                                treeGrid.putObjectAt(t.getX(), t.getY(), null);
                                //maturePlantList.remove(t);
                                maturePlantList.remove(i);
                                maturePlantList.trimToSize();
                        }
            }


            for(int i = 0; i < crownList.size(); i++){
                    Plant t = (Plant)crownList.get(i);
                        if(t.getTreeAge() < t.getMaxAge()){t.step();}
                        else{
                                crownGrid.putObjectAt(t.getX(), t.getY(), null);
                                //crownList.remove(t);
                                crownList.remove(i);
                                crownList.trimToSize();
                                }
            }
}


 public void mort(ArrayList ydominatedList, Object2DGrid yTreeG, double[ ] mort, int year){

        ArrayList[ ] aa = new ArrayList[9];
        int[ ] asize = new int[9];
        int sz = ydominatedList.size();  //saplingList1 is the arraylist to save young tree in it.


        // different species saplings are saved in different arraylist
        for (int i = 0; i < sz; i++){
         Plant sp = (Plant)ydominatedList.get(i);
         int m = spNameSeries(sp);
         if (sp.getTreeAge() == year) {
```

```
                if (aa[m] == null)          aa[m] = new ArrayList();
                aa[m].add(sp);
            }
        }

        for (int i = 0; i < 9; i++) {
          try {
              if (aa[i] != null && aa[i].size() > 0) {
                  asize[i] = aa[i].size();
                  if( asize[i] > 0 ) {
                          // how many saplings are dead
                          int mortNum = (int)( asize[i] * mort[i] );
                          if( mortNum >= 1 ) {
                             int n = 0;
                             do{
                                     int id = Random.uniform.nextIntFromTo(0, aa[i].size()-1);
                                     Plant sp = (Plant)aa[i].get(id);
                                     yTreeG.putObjectAt(sp.getX(), sp.getY(), null);
                                     if(ydominatedList == maturePlantList){
                                     for(int k = 0; k < crownList.size(); k++){
                                             Plant cr = (Plant)crownList.get(k);
                                             if(cr.getX() == sp.getX()&& cr.getY() == sp.getY()){
                                                     crownGrid.putObjectAt(cr.getX(),cr.getY(), null);
                                                     crownList.remove(k);
                                             }
                                     }
                              }
                              ydominatedList.remove(ydominatedList.indexOf(sp));
                              ydominatedList.trimToSize();
                              aa[i].remove(id);
                              aa[i].trimToSize();
                              n++;
                              }while(n < mortNum);
                          }
                  }
                  aa[i].clear();
                  asize[i] = 0;
              }
          }catch(NullPointerException ne){
                          System.out.println("Null Pointer Exception is found!");
          }
          catch(IndexOutOfBoundsException indbe){
          System.out.println("Index_outof_Bounds Exception is found!");
        }
      }
}


// this part is for the crown extend more than one cell
public int chkNeighbours(double h, double l, Object2DGrid crownGrid, Object2DGrid treeGrid, Plant t){
        int n = 0;
        if( crownGrid.getObjectAt(space.getCellCol(h), space.getCellRow(l)) != null
                && treeGrid.getObjectAt(space.getCellCol(h), space.getCellRow(l)) == null){
                        n++;
        }
        if(crownGrid.getObjectAt(space.getCellCol(h), space.getCellRow(l)) != null
                && treeGrid.getObjectAt(space.getCellCol(h), space.getCellRow(l)) != null){
                Plant t1 = (Plant)treeGrid.getObjectAt(space.getCellCol(h), space.getCellRow(l));
                double h1 = t1.getHeight();
                if(t.getHeight() <= h1){
```

```
                        n++;
                }
        }
        return n;
}


public void addCrown(double h, double l, Plant t){
        //first clean the cell

        if(crownGrid.getObjectAt(space.getCellCol(h), space.getCellRow(l)) != null //not empty
                && treeGrid.getObjectAt(space.getCellCol(h), space.getCellRow(l)) != null){//there is a big
tree in the cell
                        Plant tn = (Plant)treeGrid.getObjectAt(space.getCellCol(h), space.getCellRow(l));
                        // if the big tree is cypress pine, leave it and move it to sapling1

                        if (spTeam(tn) == 1){
                                if((tn.getSpeciesName()=="Callitris glaucophylla" ) && tn.getHeight() <
(t.getHeight())*3/5){
                                        if(sapling1.getObjectAt(space.getCellCol(h),
space.getCellRow(l))!= null){
                                                Plant yt =
(Plant)sapling1.getObjectAt(space.getCellCol(h), space.getCellRow(l));
                                                killTree(h, l, yt, saplingList1, sapling1);
                                        }
                                        //sapling1.putObjectAt(space.getCellCol(h),
space.getCellRow(l), tn);
                                        //saplingList1.add(tn);
                                        addTree(h, l, tn, saplingList1, sapling1);
                                }
                        }
                        else{
                                if(sapling1.getObjectAt(space.getCellCol(h), space.getCellRow(l))!=
null){
                                        Plant sp = (Plant)sapling1.getObjectAt(space.getCellCol(h),
space.getCellRow(l));
                                        killTree(h, l, sp, saplingList1,sapling1);
                                }
                                addTree(h, l, tn, saplingList1, sapling1);
                        }
                removeCrownAgent(h, l, tn, crownGrid);
                removeAgent(h, l, tn);
        } //clear the cell if it is occupied by a tree

        if(sapling2.getObjectAt(space.getCellCol(h), space.getCellRow(l)) != null ){
                Plant yt2 = (Plant)sapling2.getObjectAt(space.getCellCol(h), space.getCellRow(l));
                if(sapling1.getObjectAt(space.getCellCol(h), space.getCellRow(l)) != null ){
                        Plant yt1 = (Plant)sapling1.getObjectAt(space.getCellCol(h),
space.getCellRow(l));
                        if(spTeam(yt1) == 0){
                                killTree(h, l, yt1, saplingList1, sapling1);
                                addTree(h, l, yt2, saplingList1, sapling1);
                        }
                }
                killTree(h, l, (Plant)sapling2.getObjectAt(space.getCellCol(h), space.getCellRow(l)),
saplingList2, sapling2);
        }
        //add crown on the cell
        if(space.getValueAt(h, l) != 5 && space.getValueAt(h, l) != 1){
                        Plant tc = new Plant(h, l, t.getTreeAge(), t.getSpeciesName(),
```

```
                        t.getColor(),space);
                        //System.out.println(tc.getHeight());
                        tc.setGrowRate(t.getGrowRate());
                        tc.setMaxAge(t.getMaxAge());
                        tc.setHeightMax(t.getHeightMax());
                        tc.setHeight(t.getHeight());

                        crownGrid.putObjectAt(space.getCellCol(h), space.getCellRow(l), tc);
                        crownList.add(tc);
                        //dominatedList.add(tc);
                }
        }


public void randomSpring(double x, double y, Object2DGrid g, ArrayList a){
  int randomSeed = Random.uniform.nextIntFromTo(0, 10);
  if (randomSeed < 2){
        spring(x, y, g, a);
  }
}

public void addAgent(double x, double y, Plant yt) {
  //Tree t = yt;
         Plant t = new Plant(x, y, yt.getTreeAge(), yt.getSpeciesName(),
                        yt.getColor(),space);
  t.setSpeciesName(yt.getSpeciesName());
  t.setMaxAge(yt.getMaxAge());
  t.setTreeAge(yt.getTreeAge());
  t.setAgeLevel(yt.getAgeLevel());
  t.setHeight(yt.getHeight());
  t.setHeightMax(yt.getHeightMax());
        t.setGrowRate(yt.getGrowRate());
  t.setSeedingPeriod(yt.getSeedingPeriod());
  t.setSeedingRadius(yt.getSeedingRadius());
  t.setSeedsPerCell(yt.getSeedsPerCell());
        treeGrid.putObjectAt(space.getCellCol(x), space.getCellRow(y), t);
  maturePlantList.add(t);
}

public void addCrownAgent(double x, double y, Plant yt, Object2DGrid crownGrid){
        Plant tc = new Plant(x, y, yt.getTreeAge(), yt.getSpeciesName(),
                        yt.getColor(),space);
        //tc.setSpeciesName(yt.getSpeciesName());
        //Tree tc = yt;
        tc.setSpeciesName(yt.getSpeciesName());
        tc.setHeight(yt.getHeight());
        tc.setMaxAge(yt.getMaxAge());
        tc.setTreeAge(yt.getTreeAge());
        tc.setGrowRate(yt.getGrowRate());
        tc.setHeightMax(yt.getHeightMax());
        crownGrid.putObjectAt(space.getCellCol(x), space.getCellRow(y), tc);
        crownList.add(tc);
}

public void removeCrownAgent(double x, double y, Plant t, Object2DGrid crg){
        int c = t.getX();
        int r = t.getY();

        String name = t.getSpeciesName();
        double age = t.getTreeAge();
```

```
            for(int ind = 0; ind < crownList.size(); ind++){
                        Plant yt = (Plant)crownList.get(ind);
                        int age1 = yt.getTreeAge();
                        String spe = yt.getSpeciesName();
                        if (yt.getX() == c && yt.getY() == r){
                                if(age1 == age && spe == name){
                                        crg.putObjectAt(space.getCellCol(x), space.getCellRow(y), null);
                                        crownList.remove(ind);
                                        crownList.trimToSize();
                                }
                        }
                        else continue;
            }
  }

  public void removeAgent(double x, double y, Plant tree) {
   //maturePlantList.remove(maturePlantList.indexOf(tree));
            int c = tree.getX();
            int r = tree.getY();

            String name = tree.getSpeciesName();
            double age = tree.getTreeAge();
            for(int ind = 0; ind < maturePlantList.size(); ind++){
                        Plant yt = (Plant)maturePlantList.get(ind);
                        int age1 = yt.getTreeAge();
                        String spe = yt.getSpeciesName();
                        if (yt.getX() == c && yt.getY() == r){
                                if(age1 == age && spe == name){
                                        treeGrid.putObjectAt(space.getCellCol(x), space.getCellRow(y),
null);
                                        maturePlantList.remove(ind);
                                        maturePlantList.trimToSize();
                                }
                        }
                        else continue;
            }
  }

  public void killTree( double x, double y, Plant yt, ArrayList tl, Object2DGrid g) {
            int c = yt.getX();
            int r = yt.getY();

            String name = yt.getSpeciesName();
            double age = yt.getTreeAge();
            for(int ind = 0; ind < tl.size(); ind++){
                        Plant yt1 = (Plant)tl.get(ind);
                        int age1 = yt1.getTreeAge();
                        String spe = yt1.getSpeciesName();
                        if (yt1.getX() == c && yt1.getY() == r){
                                if(age1 == age && spe == name){
                                        g.putObjectAt(space.getCellCol(x), space.getCellRow(y), null);
                                        tl.remove(ind);
                                        tl.trimToSize();
                                }
                        }

            }
}

  public void replaceTree( double x, double y, Plant tree, ArrayList tl, Object2DGrid g) {
```

```
        if(g.getObjectAt(space.getCellCol(x), space.getCellRow(y)) != null){
                Plant t = (Plant)g.getObjectAt(space.getCellCol(x), space.getCellRow(y));
                int i = tl.indexOf(t);
                tl.set(i,tree);
        }

}

public void addTree(double x, double y, Plant yt, ArrayList tl, Object2DGrid g) {
                Plant t = new Plant(x, y, yt.getTreeAge(), yt.getSpeciesName(),
                                yt.getColor(),space);
        t.setSpeciesName(yt.getSpeciesName());
        t.setMaxAge(yt.getMaxAge());
        t.setTreeAge(yt.getTreeAge());
        t.setAgeLevel(yt.getAgeLevel());
        t.setHeight(yt.getHeight());
        t.setHeightMax(yt.getHeightMax());
                t.setGrowRate(yt.getGrowRate());
        t.setSeedingPeriod(yt.getSeedingPeriod());
        t.setSeedingRadius(yt.getSeedingRadius());
        t.setSeedsPerCell(yt.getSeedsPerCell());
                g.putObjectAt(space.getCellCol(x), space.getCellRow(y), t);
        tl.add(t);
        }


public int spNameSeries(Plant t){
  int k;
  k = 0;
  if(t.getSpeciesName() == "E. pilligaensis" )          { k = 0;};
  if(t.getSpeciesName() == "E. crebra" )                        { k = 1;};
  if(t.getSpeciesName() == "E. albens" )                        { k = 2;};
  if(t.getSpeciesName() == "E. populnea")                       { k = 3;};
  if(t.getSpeciesName() == "Casuarina cristata")        { k = 4;};
  if(t.getSpeciesName() == "Callitris glaucophylla") { k = 5;};
  if(t.getSpeciesName() == "Acacia deanei" )            { k = 6;};
  if(t.getSpeciesName() == "Senna nemophila")                   { k = 7;};
  if(t.getSpeciesName() == "Dodonaea viscosa")          { k = 8;};
    return k;
}

public int spTeam(Plant t){
  int k;
  String name = (String)t.getSpeciesName();
  if(name == "Acacia deanei" ||name == "Senna nemophila" || name == "Dodonaea viscosa"  )
    { k = 0; }
  else
    { k = 1; }
  return k;
}


public void clearGrid(Object2DGrid g, Vector v){
  for(int n = 0; n < v.size(); n++){
          Integer num = (Integer)v.get(n);
          int cellnum = num.intValue();
          int row = (int)cellnum/space.getSizeX();
          int col = cellnum - row * space.getSizeX();
          g.putObjectAt(col, row, null);
  }
```

```
}

public void spread(int k, double percent, Vector v, Object2DGrid g){

        int cel, cellnum;
        int r, c;
        for(int n = 1; n <= (int)(v.size() * percent); n++ ) {
                do{
                        cel = Random.uniform.nextIntFromTo(0, v.size()-1);  //get a cell index from vector
randomly
                        Integer num = (Integer)v.get(cel);
                        cellnum = num.intValue();   // Object changed as integer number.
                        r = (int)cellnum/space.getSizeX();
                        c = cellnum - r * space.getSizeX();
                }while(g.getObjectAt(c, r) != null );//||g.getValueAt(c, r) >= maxSeeds);

                //check maxSeeds.
                if(seedsPerCell[k] <= maxSeeds - numSeedsAt[cellnum]){//(int)(g.getValueAt(c, r))){
                        numSpecAt[cellnum][k] += seedsPerCell[k];
                        numSeedsAt[cellnum] += seedsPerCell[k];
                }
                else{
                        numSpecAt[cellnum][k] = numSpecAt[cellnum][k] +(maxSeeds -
numSeedsAt[cellnum]); //g.getValueAt(c, r));
                        numSeedsAt[cellnum] += maxSeeds - numSeedsAt[cellnum];
                }
                Species sp = new Species(speciesName[k], speciesColor[k], g, space);
                g.putObjectAt(c, r, sp);
        }
}



//spray uniformly
public void spreadSeeds(int k, double percent, Vector v, Object2DGrid g){

        int cel, cellnum;
        int r, c;
        for(int n = 1; n <= (int)(v.size() * percent); n++ ) {
                do{
                        cel = Random.uniform.nextIntFromTo(0, v.size()-1);  //get a cell index from vector
randomly
                        Integer num = (Integer)v.get(cel);
                        cellnum = num.intValue();   // Object changed as integer number.
                        r = (int)cellnum/space.getSizeX();
                        c = cellnum - r * space.getSizeX();
                }while(g.getObjectAt(c, r) != null );//||g.getValueAt(c, r) >= maxSeeds);

                //check maxSeeds.
                if(seedsSpray[k] <= maxSeeds - numSeedsAt[cellnum]){//(int)(g.getValueAt(c, r))){
                        numSpecAt[cellnum][k] += seedsSpray[k];
                        numSeedsAt[cellnum] += seedsSpray[k];
                }
                else{
                        numSpecAt[cellnum][k] = numSpecAt[cellnum][k] +(maxSeeds -
numSeedsAt[cellnum]); //g.getValueAt(c, r));
                        numSeedsAt[cellnum] += maxSeeds - numSeedsAt[cellnum];
                }
                Species sp = new Species(speciesName[k], speciesColor[k], g, space);
```

```
                g.putObjectAt(c, r, sp);
            }
    }

    public void population (){//, ArrayList l3){
                sp1 = 0; sp2 = 0; sp3= 0; sp4=0; sp5=0;sp6=0;sp7=0;sp8=0;
            int[ ] popT = new int[9];
            int[ ] pop = new int[9];
            int[ ] spp = new int[9];
            int k;

            for(int n = 0; n < speciesNumber; n++){
                popT[n] = 0;
            }

            for(int n = 0; n < speciesNumber; n++){
                pop[n] = 0;
            }
                for (int i = 0; i < crownList.size(); i++){
                    Plant t = (Plant)crownList.get(i);
                    k = spNameSeries(t);
                }
                for (int i = 0; i < maturePlantList.size(); i++){
                    Plant t = (Plant)maturePlantList.get(i);
                    k = spNameSeries(t);
                    popT[k]++;
                }

                for (int i = 0; i < saplingList2.size(); i++){
                    Plant t = (Plant)saplingList2.get(i);
                    k = spNameSeries(t);
                    spp[k]++;
                }

                for (int i = 0; i < saplingList1.size(); i++){
                    Plant t = (Plant)saplingList1.get(i);
                    k = spNameSeries(t);
                  pop[k]++;

                }

                        sp1 = popT[0]+pop[0]+spp[0];
                        sp2 = popT[1]+pop[1]+spp[1];
                        sp3 = popT[2]+pop[2]+spp[2];
                        sp4 = popT[3]+pop[3]+spp[3];
                        sp5 = popT[4]+pop[4]+spp[4];
                        sp6 = popT[5]+pop[5]+spp[5];
                        sp7 = popT[6]+pop[6] + spp[6];
                        sp8 = popT[7]+pop[7] + spp[7];
                        sp9 = popT[8]+pop[8] + spp[8];
                        sp10 = sp1 + sp2 + sp3 + sp4 + sp5 + sp6;
                        sp11 = sp7 + sp8 + sp9;

}

public void buildSchedule() {
  schedule.scheduleActionBeginning(1, this, "step");
  schedule.scheduleActionAtEnd(recorder1, "writeToFile");
  schedule.scheduleActionAtEnd(recorder2, "writeToFile");
  schedule.scheduleActionAtEnd(recorder3, "writeToFile");
```

```
  schedule.scheduleActionAtEnd(recorder4, "writeToFile");
        schedule.scheduleActionAtEnd(recorder5, "writeToFile");
  schedule.scheduleActionAtEnd(recorder6, "writeToFile");
  schedule.scheduleActionAtEnd(recorder7, "writeToFile");
  schedule.scheduleActionAtEnd(recorder8, "writeToFile");
  schedule.scheduleActionAtEnd(recorder9, "writeToFile");
  schedule.scheduleActionAtEnd(recorder10, "writeToFile");
  schedule.scheduleActionAtEnd(recorder11, "writeToFile");
}



public void begin() {
 buildModel();
 buildDisplay();
 buildSchedule();
        dsurf.display();
 y2surf.display();


// fsurf.display();
        tsf.display();
        shrub.display();
        first.display();
        second.display();
        third.display();
        fourth.display();
        fifth.display();

 graph.display();
}

//  setup() method to re-set all the parameter of the model
public void setup() {

 if (dsurf != null){
   dsurf.dispose();
   dsurf = null;
 }
        //if (tsurf != null) {tsurf.dispose(); tsurf = null;}
if (y2surf != null) {
          y2surf.dispose();
          y2surf = null;
        };


// if (fsurf != null) fsurf.dispose();
if (tsf != null) {tsf.dispose();tsf = null;}
if (shrub != null) {shrub.dispose();shrub = null;}
if (first != null)  {first.dispose();first = null;}
if (second != null) {second.dispose();second = null;}
if (third != null)  {third.dispose();third = null;}
if (fourth != null) {fourth.dispose(); fourth = null;}
if (fifth != null)  {fifth.dispose();fifth = null;}



 if (graph != null){
   graph.dispose();
   graph = null;
```

```
    }

    schedule = null;
    space = null;
    seedGrid = null;
    treeGrid = null;
    sapling1 = null;
    sapling2 = null;
    maturePlantList = new ArrayList();
    saplingList1 = new ArrayList();
    saplingList2 = new ArrayList();

    System.gc();
            dsurf = new DisplaySurface(this, "Vegetation Succession Model");
            registerDisplaySurface("Vegetation Succession Model", dsurf);
  //  tsurf = new DisplaySurface(this, "yTree1");
            // registerDisplaySurface("yTree1", tsurf);
    y2surf = new DisplaySurface(this, "y 2");

    registerDisplaySurface("y 2", y2surf);

            //fsurf = new DisplaySurface(this, "Dominate plants");
            //registerDisplaySurface("Dominate plants", fsurf);
            tsf = new DisplaySurface(this, "Trees Colonisation");
            registerDisplaySurface("Trees Colonisation", tsf);

            shrub = new DisplaySurface(this, "Shrubs Colonisation");
            registerDisplaySurface("Shrubs Colonisation", shrub);


            first = new DisplaySurface(this, "Height is higher than 20m");
            registerDisplaySurface ("Height is higher than 20m", first);
            second = new DisplaySurface(this, "15-20m");//"Height 2-15m");//
            registerDisplaySurface ("15-20m", second);//"Height 2-15m", second);//
            third = new DisplaySurface(this, "10-15m");//"Height lower than 2m");//
            registerDisplaySurface("10-15m", third);// "Height lower than 2m", third);
            fourth = new DisplaySurface(this, "5-10m");
            registerDisplaySurface("5-10m", fourth);
            fifth = new DisplaySurface(this, "lower than 5m");
            registerDisplaySurface("lower than 5m", fifth);

    schedule = new Schedule(1);
    maxSeeds = 500;
    speciesNumber = 9;
    lifespan = 20;

    generateNewSeed();
    Random.createUniform();
  }

  public Schedule getSchedule() {
    return schedule;
  }
  public String getName() {
    return "Vegetation Colonisation after Open-cut Mining, by X.F.Su";
  }
  public String[ ] getInitParam() {
    String[ ] params =  {"XSize", "YSize", "MaxSeeds", "SpeciesNumber", "lifespan"};
    return params;
  }
```

```
//graph.setYIncrement(200);


graph.addSequence("E. pilligaensis", new Sequence() {
  public double getSValue(){
     return getSp1();
  }
});

graph.addSequence("E. crebra", new Sequence() {
  public double getSValue(){
     return getSp2();
  }
});

graph.addSequence("E. albens", new Sequence() {
  public double getSValue(){
     return getSp3();
  }
});

graph.addSequence("E. populnea", new Sequence() {
  public double getSValue(){
     return getSp4();
  }
});
graph.addSequence("Casuarina cristata", new Sequence() {
  public double getSValue(){


    return getSp5();
  }
});
graph.addSequence("Callitris glaucophylla", new Sequence() {
        public double getSValue(){
                return getSp6();
        }
});

graph.addSequence("Acacia deanei", new Sequence() {
  public double getSValue(){
     return getSp7();
  }
});

graph.addSequence("Senna nemophila", new Sequence() {
  public double getSValue(){
     return getSp8();
  }
});

graph.addSequence("Dodonaea viscosa", new Sequence() {
  public double getSValue(){
     return getSp9();
  }
});
graph.addSequence("Total Trees", new Sequence() {
          public double getSValue(){
             return getSp10();
          }
```

```
  public int getMaxSeeds() {
    return maxSeeds;
  }
  public void setMaxSeeds(int val) {
    maxSeeds = val;
  }
  public int getSpeciesNumber() {
    return speciesNumber;
  }
  public void setLifespan(int life_span) {
      lifespan = life_span;
  }
  public int getLifespan() {
      return lifespan;
  }
  public int getSp1(){
    return sp1;
  }
  public int getSp2(){
    return sp2;
  }
  public int getSp3(){
    return sp3;
  }
  public int getSp4(){
    return sp4;
  }
  public int getSp5(){
    return sp5;
  }
  public int getSp6(){
    return sp6;
  }
  public int getSp7(){
    return sp7;
  }
  public int getSp8(){
    return sp8;
  }
  public int getSp9(){
    return sp9;
  }
  public int getSp10(){
            return sp10;
          }
  public int getSp11(){
            return sp11;
          }


  //*************************************//
  public static void main(String[ ] args) {
    // uchicago.src.sim.engine.SimInit init = new uchicago.src.sim.engine.SimInit();
    SimInit init = new SimInit();
    ForestModel model = new ForestModel();
    init.loadModel(model, null, false);
  }
}
```